Algoritmos Evolucionários em Optimização Uni e Multi-objectivo

Lino António Antunes Fernandes da Costa

Universidade do Minho
Braga, Março de 2003

Algoritmos Evolucionários em Optimização Uni e Multi-objectivo

Lino António Antunes Fernandes da Costa

Dissertação submetida à Universidade do Minho
para obtenção do grau de doutor no
Ramo de Engenharia de Produção e Sistemas,
Área de Métodos Numéricos e Estatísticos,
elaborada sob a orientação do
Professor Doutor Pedro Nuno Oliveira

Departamento de Produção e Sistemas
Escola de Engenharia
Universidade do Minho
Braga, Março de 2003

Resumo

Nesta dissertação, as duas abordagens evolucionárias mais conhecidas e frequentemente aplicadas a problemas de optimização são consideradas: as Estratégias Evolutivas e os Algoritmos Genéticos. Estas abordagens apresentam como principal desvantagem tempos de computação elevados mas, por outro lado, exibem uma grande flexibilidade na modulação de problemas de engenharia. As vantagens e os inconvenientes de cada uma das abordagens para diferentes tipos de problemas de optimização foram investigadas, nomeadamente ao nível da representação do espaço das soluções e no tratamento das restrições. Usualmente, em engenharia, os problemas são formulados com um único objectivo, em geral, o custo. Contudo, podem existir outros objectivos que, quando considerados, transformam o problema em optimização multi-objectivo. Nesta classe de problemas, em geral, existem múltiplos objectivos conflituosos, que fazem com que exista um conjunto de soluções compromisso. As abordagens evolucionárias, ao trabalharem com populações de pontos, parecem ser úteis na resolução de problemas multi-objectivo mas, apesar do sucesso, o elitismo surgiu como uma forma eficiente de melhorar o desempenho em optimização multi-objectivo. Neste trabalho, um novo esquema de elitismo, através do qual é possível controlar o tamanho da população elitista, bem como a concentração de pontos eficientes do problema multi-objectivo, é introduzido. Uma vez que quase todas as abordagens evolucionárias para optimização multi-objectivo são baseadas em Algoritmos Genéticos e implementações baseadas em Estratégias Evolutivas são muito raras, uma nova abordagem, baseada em Estratégias Evolutivas, é proposta. Diversos mecanismos, como o elitismo, um esquema de *sharing* adaptativo e uma selecção geométrica, foram introduzidos no *Multiobjective Elitist Evolution Strategy* para melhorar o seu desempenho.

Diversas aplicações de Algoritmos Evolucionários a problemas de engenharia foram realizadas. Em primeiro lugar, os Algoritmos Evolucionários foram aplicados a problemas de programação inteira mista não linear da área de engenharia química. Um Algoritmo Genético foi também aplicado ao problema de optimização de placas laminadas. Estes problemas de optimização de placas foram, numa primeira fase, formulados como problemas de programação inteira mista com restrições com uma única função objectivo. Numa segunda fase, as formulações multi-objectivo alternativas dos problemas foram também resolvidas. O algoritmo multi-objectivo desenvolvido foi comparado com outras abordagens evolucionárias para optimização multi-objectivo em diversos problemas teste, tendo-se verificado que o esquema elitista desenvolvido prova ser um bom compromisso entre o tempo computacional requerido e o tamanho da população elitista.

Abstract

In this dissertation, the two best known evolutionary approaches and often applied to optimization problemas are considered: the Evolution Strategies and the Genetic Algorithms. These approaches present as main disadvantage high computational times but, on the other hand, exhibit a great flexibility in modelling engineering problems. The advantages and the drawbacks of each approach to distinct kind of optimization problems are investigated, in particular, representation issues and constraint handling. Usually, in engineering, problems are formulated with a single objective, in general, the cost. However, other objectives might exist that, when considered, transform the problem in multi-objective optimization. In the latter class of problems, usually, there are multiple conflicting objectives, giving rise to a set of compromise solutions. The evolutionary approaches, population based, seem to be useful in tackling multi-objective problems. However, in spite of its success, elitism has emerged as an effective way of improving the performance of the multi-objective evolutionary algorithms. In this work, a new elitist scheme, by which it is possible to control the size of the elite population, as well as, the concentration of points approximating the efficient solutions of the multi-objective problem, is introduced. Almost all approaches to multi-objective optimization are based on Genetic Algorithms, and implementations based on Evolution Strategies are very rare. Thus, a new evolutionary approach to multi-objective optimization, based on Evolution Strategies, is proposed. Several mechanisms, like elitism, an adaptive sharing scheme and a geometric selection, have been introduced in the Multi-objective Elitist Evolution Strategy in order to improve the algorithm performance.

Several applications of Evolutionary Algorithms to distinct engineering problemas were implemented. Firstly, the Evolutionary Algorithms were applied to the global optimization of mixed integer non-linear problems of the area of chemical engineering. A Genetic Algorithm was also applied to laminated plate optimization problems. These optimization plate problems are firstly formulated as a constrained mixed-integer programming problem with a single objective function. The alternative multi-objective formulations of the problems were also solved. The proposed multi-objective algorithm was compared with other evolutionary multi-objective algorithms in several test problems. The new elitist scheme proves to lead to a good compromise between computational time and size of the elite population.

À Sílvia, Mariana e Lino.

Agradecimentos

A realização desta dissertação só foi possível graças ao apoio de várias pessoas e instituições. Gostaria de expressar o meu reconhecimento e gratidão a algumas pessoas em especial.

Ao Professor Pedro Nuno Oliveira, meu orientador, pelas suas inúmeras sugestões, constante incentivo e precioso apoio na execução deste trabalho.

À Fundação Calouste Gulbenkian e à Fundação Luso-Americana para o Desenvolvimento, agradeço as bolsas concedidas que me apoiaram as deslocações aos EUA. Devo mencionar que parte deste trabalho teve também o apoio do programa PRAXIS, do Ministério da Ciência e Tecnologia ao projecto "*Modelos Variacionais e Optimização* " (PRAXIS/PCEX/P/MAT/38/96). No âmbito deste projecto, destaco a colaboração do Professor Rogério Leal do Departamento de Mecânica da Universidade de Coimbra e da Professora Isabel Figueiredo do Departamento de Matemática da Universidade de Coimbra que permitiu a realização de parte deste trabalho.

Estou também grato aos meus colegas do Departamento de Produção e Sistemas, em especial, aos do sub-grupo de Métodos Numéricos e Estatísticos por todo o apoio que manifestaram.

Finalmente, agradeço à Sílvia pela sua dedicação, paciência e incentivo, e aos meus pais pelo seu constante apoio. Sem eles este trabalho nunca teria sido realizado.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

AE	Algoritmo Evolucionário
AG	Algoritmo Genético
CE	Computação Evolucionária
EE	Estratégia Evolutiva
OM	Optimização Multi-objectivo
OU	Optimização Uni-objectivo
PE	Programação Evolucionária
PG	Programação Genética
PINL	Programação Inteira Não Linear
PIMNL	Programação Inteira Mista Não Linear
PL	Programação Linear
PNL	Programação Não Linear
PS	População Secundária
MEGA	*Multiobjective Elitist Genetic Algorithm*
MEES	*Multiobjective Elitist Evolution Strategy*

Lista de Símbolos

$\mathbf{x}$	Vector das variáveis de decisão contínuas
n	Número de variáveis de decisão contínuas
$f(\mathbf{x})$	Função objectivo (variáveis de decisão contínuas)
Ω	Domínio de $f(\mathbf{x})$ ($\Omega \subseteq \Re^n$)
$\mathbf{g}(\mathbf{x})$	Vector das restrições do tipo desigualdade (variáveis de decisão contínuas)
m	Número de restrições do tipo desigualdade
$\mathbf{h}(\mathbf{x})$	Vector das restrições do tipo igualdade (variáveis de decisão contínuas)
p	Número de restrições do tipo igualdade
$\underline{x_k}$	Limite inferior da variável de decisão contínua k
$\overline{x_k}$	Limite superior da variável de decisão contínua k
$\mathbf{x}^*$	Ponto óptimo
A	Região admissível
$R(\mathbf{x})$	Conjunto de restrições activas no ponto admissível $\mathbf{x}$
$\mathbf{y}$	Vector das variáveis de decisão inteiras
q	Número de variáveis de decisão inteiras
$f(\mathbf{y})$	Função objectivo (variáveis de decisão inteiras)
Ψ	Domínio de $f(\mathbf{y})$ ($\Psi \subseteq Z^n$)
$\mathbf{g}(\mathbf{y})$	Vector das restrições do tipo desigualdade (variáveis de decisão inteiras)
$\mathbf{h}(\mathbf{y})$	Vector das restrições do tipo igualdade (variáveis de decisão inteiras)
$\underline{y_k}$	Limite inferior da variável de decisão inteira k
$\overline{y_k}$	Limite superior da variável de decisão inteira k
$f(\mathbf{x}, \mathbf{y})$	Função objectivo (variáveis de decisão contínuas e inteiras)
$\mathbf{g}(\mathbf{x}, \mathbf{y})$	Vector das restrições do tipo desigualdade (variáveis de decisão contínuas e inteiras)
$\mathbf{h}(\mathbf{x}, \mathbf{y})$	Vector das restrições do tipo igualdade (variáveis de decisão contínuas e inteiras)
$\nabla f(\mathbf{x})$	Vector gradiente da função objectivo (variáveis de decisão contínuas)

$\nabla^2 f(\mathbf{x})$	Matriz hessiana da função objectivo (variáveis de decisão contínuas)
$\nabla g_j(\mathbf{x})$	Vector gradiente da restrição do tipo desigualdade j (variáveis de decisão contínuas)
$\nabla h_i(\mathbf{x})$	Vector gradiente da restrição do tipo igualdade i (variáveis de decisão contínuas)
$L(\mathbf{x}, \lambda)$	Função lagrangeana (variáveis de decisão contínuas)
λ	Vector dos multiplicadores de Lagrange
$\nabla L(\mathbf{x}, \lambda)$	Vector gradiente da função lagrangeana (variáveis de decisão contínuas)
$\nabla^2 L(\mathbf{x}, \lambda)$	Matriz hessiana da função lagrangeana (variáveis de decisão contínuas)
Φ	Espaço dos indivíduos (Algoritmos Evolucionários)
$F(\mathbf{x})$	Função medida do desempenho (Algoritmos Evolucionários)
μ	Tamanho da população de progenitores (Estratégias Evolutivas)
λ	Tamanho da população de descendentes (Estratégias Evolutivas)
P	Tamanho da população principal (Algoritmos Genéticos)
l	Comprimento do cromossoma (Algoritmos Genéticos)
$\mathbf{f}(\mathbf{x})$	Vector das funções objectivo (variáveis de decisão contínuas)
s	Número de funções objectivo
θ	Tamanho da elite
d	Distância desejável entre soluções na população secundária
S	Tamanho máximo da população secundária
Q_k	Quantidade de soluções na frente k
r	Razão de variação de Q_f
ps	Pressão de selecção

Capítulo 1

Introdução

Este capítulo começa por clarificar os motivos que levaram à realização dos trabalhos descritos ao longo desta dissertação. Em seguida, a sua organização geral é sumariamente apresentada, bem como os respectivos contributos.

1.1 Motivação

De acordo com a Teoria Evolucionista de Darwin, a evolução dos seres vivos ocorre devido à selecção natural e à adaptação ao ambiente. A selecção natural faz com que os seres vivos mais aptos, em relação ao ambiente, tenham maior probabilidade de sobrevivência. Por outro lado, para que seja possível a adaptação dos seres vivos a um ambiente continuamente em mudança, na natureza surgiram mecanismos que tornam possível a diversidade, i.e., a existência de seres vivos com características próprias que, potencialmente, os podem tornar mais aptos. O processo evolutivo das espécies pode ser visto como um processo de optimização consistindo na adaptação dos seres vivos ao meio ambiente. Este modelo biológico serviu de inspiração ao desenvolvimento de diversos Algoritmos Evolucionários (AEs) [Rec64, Sch85, Hol75, Gol89] com o objectivo de serem aplicados a problemas de optimização. Por outro lado, existem problemas de optimização em engenharia aos quais, devido às suas características, não é fácil, ou mesmo possível, a aplicação dos algorit-

mos de optimização tradicionais. Em particular, os AEs não fazem qualquer exigência ao nível da continuidade, diferenciabilidade e convexidade, e desenvolvem a procura com base numa população de pontos. Neste sentido, importa caracterizar melhor o domínio de aplicação dos AEs, sobretudo em problemas de natureza combinatória e de grande dimensão. Portanto, é relevante, por um lado, estudar as condições de aplicabilidade dos AEs a problemas de optimização em engenharia e, por outro lado, determinar as vantagens e limitações de cada abordagem evolucionária, com eventual identificação de classes de problemas para os quais uma dada abordagem seja mais adequada. Para além disso, dada a natureza multi-objectivo de grande parte dos problemas de engenharia, interessa estudar a aplicação das abordagens evolucionárias a este tipo de problemas. Esta área tem sido objecto de vários contributos de que se destacam os trabalhos de Fonseca [Fon95], Zitzler [ZT99] e Deb [Deb01]. Estes contributos, essencialmente baseados em AGs, motivaram o desenvolvimento do presente trabalho que estendeu a aplicação de EEs à resolução de problemas multi-objectivo, tendo presente os trabalhos de Kursawe [Kur90] e Knowles e Corne [KC99].

1.2 Organização

Com o objectivo de facilitar a leitura, esta dissertação encontra-se dividida em duas partes principais. A Parte I foca a Optimização Uni-objectivo e, por outro lado, a Parte II aborda a Optimização Multi-objectivo. Uma vez que a optimização multi-objectivo envolve múltiplos objectivos, parece intuitivo poder-se considerar a optimização uni-objectivo, com um único objectivo, como um caso particular da optimização multi-objectivo. Apesar disso, a divisão da dissertação em duas partes como foi indicado é importante, uma vez que muitos dos problemas focados podem ser modelados quer como problemas uni-objectivo quer como problemas multi-objectivo. Neste sentido, interessa descrever os fundamentos,

as técnicas e as aplicações nas duas perspectivas. Um outro argumento prende-se com o facto de que grande parte do esforço de investigação na área de optimização está relacionado com a optimização uni-objectivo e, a optimização multi-objectivo ser encarada como uma aplicação da optimização uni-objectivo para manusear diversos objectivos.

Na Parte I, na perspectiva de optimização uni-objectivo, começa-se por definir as formulações dos problemas e abordar alguns conceitos básicos. Em seguida, são descritas diferentes técnicas evolucionárias utilizadas em optimização. Finalmente, diversas aplicações destas técnicas a problemas da área de engenharia são apresentadas.

A Parte II é iniciada com a apresentação de diversos conceitos básicos de optimização multi-objectivo. Duas abordagens à resolução de problemas multi-objectivo baseadas em duas técnicas evolucionárias distintas são propostas. Finalmente, algumas aplicações dos algoritmos desenvolvidos a problemas da área de engenharia são apresentadas.

No final da dissertação, para além das propostas de realização de trabalho futuro, são elaboradas algumas conclusões relativas ao trabalho realizado.

1.3 Contribuições

As principais contribuições deste trabalho são:

- Comparação empírica das Estratégias Evolutivas e dos Algoritmos Genéticos quando aplicados a problemas de Programação Não Linear uni-objectivo com e sem restrições [CO98];
- Aplicação de Estratégias Evolutivas e Algoritmos Genéticos a problemas de Programação Inteira Mista Não Linear uni-objectivo da área de engenharia química [CO99, CO01]. Comparações dos algoritmos evolucionários com o algoritmo M-SIMPSA baseado na combinação do método simplex de Nelder-Mead e *Simulated Annealing* são realizadas;

- Aplicação de Algoritmos Genéticos à optimização de placas laminadas com materiais isotrópicos e compósitos, quer formulados como problemas de Programação Inteira Mista Não Linear com Restrições com um único objectivo [FFJ$^+$98, COIL99, COI$^+$00, LCOF00], quer formulados como problemas multi-objectivo [COF$^+$02];

- Um novo esquema de elitismo baseado numa população secundária incorporando mecanismos de controlo do nível de elitismo e da concentração de soluções na população secundária. O *Multiobjective Elitist Genetic Algorithm*, integrando o novo esquema elitista, é aplicado a um conjunto de problemas de optimização multi-objectivo com a finalidade de investigar o efeito do nível de elitismo e do parâmetro de controlo da concentração de pontos na população secundária [CO02a];

- Um novo algoritmo para optimização multi-objectivo, *Multiobjective Elitist Evolution Strategy*, adoptando as principais características das Estratégias Evolutivas, mas utilizando elitismo, um esquema adaptativo de *sharing* e selecção geométrica. O efeito de diversos parâmetros do algoritmo é estudado para um conjunto de problemas multi-objectivo. Comparações com outras abordagens evolucionárias para optimização multi-objectivo são realizadas [CO02b];

- Desenvolvimento de aplicações informáticas resultantes da implementação das Estratégias Evolutivas e Algoritmos Genéticos para optimização uni-objectivo e do *Multiobjective Elitist Evolution Strategy* e do *Multiobjective Elitist Genetic Algorithm* para optimização multi-objectivo.

Parte I

Optimização Uni-objectivo

Capítulo 2

Introdução

Esta primeira parte relativa a Optimização Uni-objectivo (OU) é iniciada com um capítulo que pretende definir os tipos de problemas para os quais se vai investigar a aplicação de Algoritmos Evolucionários (AEs). Assim, na Secção 2.1 é apresentada a formulação matemática de um problema de optimização com um objectivo (uni-objectivo), bem como diversos conceitos básicos, mas essenciais à compreensão da dissertação. Em seguida, na Secção 2.2 abordam-se, genericamente, as diferentes técnicas evolucionárias, destacando-se as suas potencialidades na resolução de problemas de optimização uni-objectivo. Nesta secção são apresentadas as características dos AEs por forma a permitir a identificação das diferenças entre estas abordagens e os algoritmos de optimização tradicionais. Para além disso, são indicadas algumas das áreas de engenharia em que estes algoritmos têm sido aplicados.

Uma vez que para o objectivo do presente trabalho interessa o estudo da aplicação dos AEs na resolução de problemas de optimização, apesar da grande diversidade de AEs existentes, apenas os dois mais frequentemente utilizados neste tipo de problemas são considerados. Deste modo, os AEs desenvolvidos pelas duas principais escolas de computação evolucionária são apresentados nas Secções 3.1 e 3.2 do Capítulo 3. Nestas secções descrevem-se, com detalhe, no contexto da optimização, as Estratégias Evolutivas (EEs) [Rec73, Sch95] e os Algoritmos Genéticos (AGs) [Hol75, Gol89]. De notar que estas duas

abordagens não são as únicas inspiradas na natureza para resolver problemas de optimização, existindo outras, tais como, a Programação Evolucionária (PE) [FOW66], a Programação Genética (PG) [Koz92] e, surgidas mais recentemente, o algoritmo de Colónia de Formigas (*Ant Colony*) [DMC96] e o *Particle Swarm Algorithm* [KE95]. Descrições pormenorizadas destas abordagens podem ser encontradas na literatura como é referido ao longo deste capítulo. Finalmente, com o objectivo de se comparar o desempenho das EEs e dos AGs, são apresentados na Secção 3.3 os resultados obtidos por estas abordagens na resolução de diversos problemas de optimização não linear.

O Capítulo 4 encerra esta primeira parte com a apresentação de algumas aplicações de AEs a problemas de optimização de diversas áreas da engenharia. Começa-se por apresentar, na Secção 4.1, a aplicação de AGs e EEs à optimização global de problemas inteiros mistos não lineares (PIMNL). Estes AEs são também comparados com um algoritmo baseado em *Simulated Annealing* (M-SIMPSA) [Car98]. De seguida, na Secção 4.2 é descrita a aplicação de um AG ao projecto de placas laminadas com materiais isotrópicos. Na última secção desta primeira parte, a Secção 4.3, descreve-se a aplicação de um AG ao projecto de placas laminadas com materiais compósitos.

2.1 Optimização Uni-objectivo

Uma vez que os AEs serão abordados na perspectiva de métodos de procura e optimização, é relevante conhecer alguns conceitos gerais de optimização (também conhecida por *Programação Matemática*). Nesta primeira parte, apenas se vão considerar problemas de optimização uni-objectivo, i.e., com uma única função objectivo. Mais adiante, quando conveniente, apresenta-se a formulação matemática de problemas multi-objectivo (com mais de uma função objectivo), bem como os conceitos que estão associados a este tipo de problemas de optimização. Nesta secção, também, são feitas algumas considerações acerca

dos algoritmos de optimização tradicionais para se destacar as características distintas que as abordagens baseadas em AEs possuem.

2.1.1 Formulação Matemática

Matematicamente, um problema de optimização com um único objectivo consiste na minimização ou maximização de uma função sujeita a restrições [NW99, NS96]. A seguinte notação irá ser considerada:

- $\mathbf{x}$ é o vector das *variáveis de decisão* (também, muitas vezes, conhecidas por *parâmetros*);
- $f(\mathbf{x})$ é a *função objectivo*, i.e., a função que se pretende maximizar ou minimizar;
- $\mathbf{g}(\mathbf{x})$ é o vector das *restrições do tipo desigualdade* que devem ser satisfeitas;
- $\mathbf{h}(\mathbf{x})$ é o vector das *restrições do tipo igualdade* que devem ser satisfeitas.

Utilizando-se esta notação, um problema de minimização pode ser formulado, genericamente, da seguinte forma:

$$\min f(\mathbf{x}) \text{ onde } \mathbf{x} \in \Omega \tag{2.1.1}$$

sujeito a

$$g_j(\mathbf{x}) \geq 0 \text{ com } j = 1, \ldots, m$$

$$h_i(\mathbf{x}) = 0 \text{ com } i = m+1, \ldots, m+p.$$

Na formulação apresentada existem n variáveis reais, m restrições do tipo desigualdade e p restrições do tipo igualdade (o número total de restrições é $m+p$). O *espaço das variáveis*, $\Omega \subseteq \Re^n$, corresponde ao conjunto de todos os valores possíveis para as variáveis de decisão. A procura da solução de um problema de optimização como o formulado, o ponto *óptimo* $\mathbf{x}^*$, é feita no espaço das variáveis. Em geral, os problemas de optimização são abordados

pressupondo-se que o ponto óptimo $\mathbf{x}^*$ existe, é único, e pode ser localizado utilizando um algoritmo de optimização. Apesar de muitas vezes este ser o caso, existem situações em que tal não se verifica: se $f(\mathbf{x})$ não é limitada inferiormente, então $\mathbf{x}^*$ não existe ou, para certas funções objectivo, $\mathbf{x}^*$ poderá não ser único. As restrições de desigualdade são expressas em termos de desigualdades do tipo maior ou igual ($\geq$)[1]. Muitas vezes, poderão existir restrições de desigualdade especificando limites inferiores ($\underline{x_k}$) e superiores ($\overline{x_k}$) para as n variáveis x_k, i.e., $\underline{x_k} \leq x_k \leq \overline{x_k}$, para $k = 1, \ldots, n$.

Qualquer ponto $\mathbf{x} \in \Omega$ diz-se satisfazer uma restrição se, para essa restrição, o lado esquerdo da expressão calculado nesse ponto está de acordo com o lado direito, em termos do operador relacional. Um ponto é dito *ponto admissível* se todas as restrições do tipo desigualdade e restrições do tipo igualdade forem satisfeitas nesse ponto. O conjunto de todos os pontos admissíveis constitui a *região admissível* e pode ser definido da seguinte forma:

$$A = \{\mathbf{x} \in \Omega : \mathbf{g}(\mathbf{x}) \geq 0 \wedge \mathbf{h}(\mathbf{x}) = 0\}.$$

Todos os pontos que não pertencem ao conjunto A são pontos não admissíveis. O óptimo $\mathbf{x}^*$, a solução do problema, pertence necessariamente à região admissível. Quando um ponto satisfaz uma restrição j do tipo desigualdade, duas situações podem ocorrer:

- o ponto está no limite da região admissível da restrição, i.e., $g_j(\mathbf{x}) = 0$ e neste caso a restrição j diz-se *activa*;
- o ponto está no interior da região admissível da restrição, i.e., $g_j(\mathbf{x}) > 0$ e neste caso a restrição j diz-se *inactiva*.

Para qualquer ponto admissível $\mathbf{x}$, todas as p restrições do tipo igualdade estão activas.

[1] Qualquer restrição expressa em termos da desigualdade menor ou igual pode ser facilmente transformada em termos da desigualdade maior ou igual bastando para isso multiplicá-la por –1.

Define-se *conjunto activo* $R(\mathbf{x})$, para qualquer ponto admissível $\mathbf{x}$, como sendo o conjunto dos índices de todas as restrições activas, i.e.,

$$R(\mathbf{x}) = \big\{j \in \{1,\ldots,m\} : g_j(\mathbf{x}) = 0\big\} \cup \big\{i \in \{m+1,\ldots,m+p\} : h_i(\mathbf{x}) = 0\big\}.$$

De notar que qualquer problema formulado em termos de maximização de uma função objectivo $f(\mathbf{x})$ pode ser reformulado da seguinte forma: $\max f(\mathbf{x}) = -\min(-f(\mathbf{x}))$.

Os problemas com a formulação geral (2.1.1) podem ser classificados de acordo, por exemplo, com a natureza da função objectivo e das restrições (e.g., linear, não linear, convexa), o número de variáveis (grande ou pequeno), a diferenciabilidade da função objectivo e restrições, a existência ou não de restrições. Os problemas em que a função objectivo e todas as restrições são funções lineares relativamente a $\mathbf{x}$ constituem os chamados problemas de *Programação Linear* (PL). Por outro lado, os problemas em que pelo menos uma das restrições ou a função objectivo não são lineares em relação a $\mathbf{x}$ são conhecidos por problemas de *Programação Não Linear* (PNL).

2.1.2 Optimização Contínua versus Optimização Discreta

O termo optimização contínua refere-se a problemas em que a procura do óptimo é feita num conjunto infinito de pontos, i.e., o espaço de procura é infinito. Nestes problemas, tipicamente, as variáveis de decisão são reais, como é o caso de todos os problemas que correspondem à formulação (2.1.1). Por outro lado, o termo optimização discreta refere-se a problemas em que a procura do óptimo se faz num conjunto de pontos.

Devido à sua natureza, para um grande número de problemas, as variáveis só fazem sentido se tomarem apenas valores inteiros. Por outro lado, a resolução de um problema deste tipo ignorando a natureza inteira das variáveis, i.e., considerando-as reais (e arredondando-as para o inteiro mais próximo) não garante a obtenção de uma solução próxima do óptimo do problema original. Estes problemas com variáveis inteiras são designados de problemas

de *Programação Inteira* [NW99, NS96] e podem ser formulados da seguinte forma:

$$\min f(\mathbf{y}) \text{ onde } \mathbf{y} \in \Psi \tag{2.1.2}$$

sujeito a

$$g_j(\mathbf{y}) \geq 0 \text{ com } j = 1, \ldots, m$$

$$h_i(\mathbf{y}) = 0 \text{ com } i = m+1, \ldots, m+p.$$

Nesta formulação, $\mathbf{y}$ é o vector das q variáveis inteiras definidas em $\Psi \subseteq Z^q$, onde Z corresponde ao conjunto de todos os inteiros. Nestes problemas, muitas vezes, são consideradas restrições de desigualdade especificando limites inferiores ($\underline{y_l}$) e superiores ($\overline{y_l}$) para as q variáveis y_l, i.e., $\underline{y_l} \leq y_l \leq \overline{y_l}$, para $l = 1, \ldots, q$.

Para além dos problemas que se podem descrever por uma das formulações anteriores (2.1.1) e (2.1.2), existem os que, devido à sua natureza, devem ser modelados utilizando quer variáveis reais quer variáveis inteiras. Estes problemas são, em geral, designados por problemas de *Programação Inteira Mista*. A formulação matemática destes problemas pode ser feita da seguinte forma:

$$\min f(\mathbf{x}, \mathbf{y}) \text{ onde } \mathbf{x} \in \Omega \text{ e } \mathbf{y} \in \Psi \tag{2.1.3}$$

sujeito a

$$g_j(\mathbf{x}, \mathbf{y}) \geq 0 \text{ com } j = 1, \ldots, m$$

$$h_i(\mathbf{x}, \mathbf{y}) = 0 \text{ com } i = m+1, \ldots, m+p.$$

Na formulação apresentada existe um total de $n+q$ variáveis, sendo $\mathbf{x}$ e $\mathbf{y}$, respectivamente, os vectores das n variáveis reais e das q variáveis inteiras.

Tal como anteriormente, relativamente às formulações (2.1.2) e (2.1.3), se, para um dado problema, pelo menos uma das restrições ou a função objectivo não forem lineares em relação a $\mathbf{x}$ ou $\mathbf{y}$, então esse problema é um problema de *Programação Inteira Não Linear* (PINL) ou um problema de *Programação Inteira Mista Não Linear* (PIMNL).

2.1.3 Optimização Local, Global e Convexidade

Em geral, pretende-se encontrar o *óptimo global*, i.e., para um problema de minimização, o ponto para o qual a função objectivo atinge o seu mínimo. Matematicamente, pode-se considerar a seguinte definição [NW99, NS96].

Definição 2.1.1. (**Mínimo Global**) *Um ponto* $\mathbf{x}^*$ *é mínimo global se* $f(\mathbf{x}^*) \leq f(\mathbf{x})$ *para todo o* $\mathbf{x} \in \Omega$.

Por outro lado, muitas vezes, os algoritmos apenas permitem encontrar um *mínimo local* que é o ponto que na sua vizinhança apresenta o menor valor da função objectivo. Define-se uma vizinhança de $\mathbf{x}$ como sendo um conjunto aberto que contém $\mathbf{x}$ e que está contido no domínio de $f(\mathbf{x})$. Formalmente, as seguintes definições podem ser consideradas.

Definição 2.1.2. (**Mínimo Local Fraco**) *Um ponto* $\mathbf{x}^*$ *é mínimo local fraco se existe uma vizinhança* V *de* $\mathbf{x}^*$ *tal que* $f(\mathbf{x}^*) \leq f(\mathbf{x})$ *para todo o* $\mathbf{x} \in V$.

Definição 2.1.3. (**Mínimo Local Forte**) *Um ponto* $\mathbf{x}^*$ *é mínimo local forte se existe uma vizinhança* V *de* $\mathbf{x}^*$ *tal que* $f(\mathbf{x}^*) < f(\mathbf{x})$ *para todo o* $\mathbf{x} \in V$ *e* $\mathbf{x} \neq \mathbf{x}^*$.

O conceito de convexidade é importante em optimização e pode ser definido para conjuntos e para funções. Em seguida define-se o conceito de convexidade.

Definição 2.1.4. (**Conjunto Convexo**) *Um conjunto* $S \in \Re^n$ *é convexo se, para quaisquer dois pontos desse conjunto,* $\mathbf{a} \in S$ *e* $\mathbf{b} \in S$, *se tem* $\alpha\mathbf{a} + (1-\alpha)\mathbf{b} \in S$ *para todo o* $\alpha \in [0,1]$ *(qualquer recta ligando quaisquer dois pontos do conjunto* S *ainda pertence a* S*).*

Definição 2.1.5. (**Função Convexa**) *Uma função* f *é convexa no seu domínio se, para quaisquer dois pontos,* $\mathbf{a}$ *e* $\mathbf{b}$, *desse domínio, se tem* $f(\alpha\mathbf{a}+(1-\alpha)\mathbf{b}) \leq \alpha f(\mathbf{a})+(1-\alpha)f(\mathbf{b})$ *para todo* $\alpha \in [0,1]$ *(graficamente, o gráfico de* f *fica abaixo da recta que une os pontos* $(\mathbf{a}, f(\mathbf{a}))$ *a* $(\mathbf{b}, f(\mathbf{b}))$ *no espaço* $\Re^{n+1}$*).*

Uma função f é dita *côncava* se $-f$ é convexa.

Em geral, os problemas de PNL com e sem restrições podem possuir óptimos locais que não são óptimos globais. No entanto, se, por exemplo, para um dado problema de optimização sem restrições, se souber que a função objectivo $f(\mathbf{x})$ é convexa em todo o seu domínio, então todo o óptimo local é também óptimo global.

O termo *Programação Convexa* é, muitas vezes, utilizado para descrever problemas de optimização com restrições em que:

- a função objectivo é convexa;
- as restrições do tipo desigualdade são côncavas;
- as restrições do tipo igualdade são lineares.

Para os problemas em que se verificam as anteriores condições, todo o óptimo local é também óptimo global (é o caso de todos os problemas de PL) [NW99].

2.1.4 Condições de Optimalidade

Nesta secção são apresentadas as condições de optimalidade para problemas de optimização como os formulados em (2.1.1) [NW99]. Contudo, por questões de clareza, em primeiro lugar, apresentam-se as condições de optimalidade apenas para problemas sem restrições. Em seguida, enunciam-se as condições para problemas de optimização com restrições do tipo desigualdade e do tipo igualdade. As demonstrações dos teoremas que serão apresentados nesta secção podem ser encontradas em [NW99].

Problemas sem Restrições

Para problemas sem restrições, quando a função objectivo $f(\mathbf{x})$ é continuamente diferenciável, é possível a identificação de mínimos locais pela análise de determinadas condições

de optimalidade. Em particular, se a função objectivo $f(\mathbf{x})$ é duas vezes continuamente diferenciável, pode-se afirmar que um ponto $\mathbf{x}^*$ é um mínimo local pela análise das suas primeiras e segundas derivadas em $\mathbf{x}^*$. Para uma dada função objectivo $f(\mathbf{x})$, o *gradiente* $\nabla f(\mathbf{x})$ e a *hessiana* $\nabla^2 f(\mathbf{x})$ são, respectivamente, o vector das primeiras derivadas parciais e a matriz das segundas derivadas parciais, i.e.,

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

e

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_2 \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_n \partial x_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \frac{\partial^2 f}{\partial x_2 \partial x_n} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}.$$

Assumindo-se que $\mathbf{x}^*$ é mínimo local, podem-se deduzir as condições necessárias de optimalidade.

Teorema 2.1.1. *(**Condição Necessária de Primeira Ordem**) Se $\mathbf{x}^*$ é um mínimo local e $f(\mathbf{x})$ é continuamente diferenciável numa vizinhança aberta de $\mathbf{x}^*$, então $\nabla f(\mathbf{x}^*) = 0$.*

Teorema 2.1.2. *(**Condição Necessária de Segunda Ordem**) Se $\mathbf{x}^*$ é um mínimo local e $\nabla^2 f(\mathbf{x})$ é contínua numa vizinhança aberta de $\mathbf{x}^*$, então $\nabla f(\mathbf{x}^*) = 0$ e $\nabla^2 f(\mathbf{x}^*)$ é semidefinida positiva.*

Qualquer ponto que satisfaça $\nabla f(\mathbf{x}) = 0$ é designado por *ponto estacionário* de $f(\mathbf{x})$. De notar que um *ponto estacionário* pode ser ou um mínimo local de $f(\mathbf{x})$, ou um máximo local de $f(\mathbf{x})$, ou um ponto que nem é mínimo nem é máximo e que é então designado por *ponto de sela* ou *ponto de descanso*. Em geral, as condições necessárias são utilizadas para mostrar que um determinado ponto não é óptimo, dado que um mínimo local deverá verificar as

condições necessárias. Por outro lado, as condições suficientes que serão enunciadas em seguida, se se verificarem num ponto $\mathbf{x}^*$, garantem que $\mathbf{x}^*$ é um mínimo local de $f(\mathbf{x})$.

Teorema 2.1.3. *(**Condições Suficientes de Segunda Ordem**) Se $\nabla^2 f(\mathbf{x})$ é contínua numa vizinhança aberta de $\mathbf{x}^*$ e se $\nabla f(\mathbf{x}^*) = 0$ e $\nabla^2 f(\mathbf{x}^*)$ é definida positiva, então $\mathbf{x}^*$ é um mínimo local forte de $f(\mathbf{x})$.*

Estas condições são quase necessárias e suficientes, não se verificando para o caso de curvatura nula.

Problemas com Restrições

Para problemas com restrições também é possível estabelecer condições de optimalidade. Estas condições serão apresentadas em termos da *função lagrangeana*. Para a formulação do problema com restrições (2.1.1) a função lagrangeana é definida por

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \sum_{j=1}^{m} \lambda_j g_j(\mathbf{x}) - \sum_{i=m+1}^{m+p} \lambda_i h_i(\mathbf{x}),$$

onde λ é o vector dos multiplicadores de Lagrange.

Para problemas com restrições é necessário analisar as propriedades dos gradientes das restrições $\nabla g_j(\mathbf{x})$ e $\nabla h_i(\mathbf{x})$ que são, respectivamente, os vectores das primeiras derivadas das m restrições do tipo desigualdade e das p restrições do tipo igualdade. Um vector gradiente duma restrição $\nabla g_j(\mathbf{x})$ (ou $\nabla h_i(\mathbf{x})$) é, em geral, perpendicular ao contorno da restrição $g_j(\mathbf{x})$ (ou $h_i(\mathbf{x})$). No caso duma restrição do tipo desigualdade, o respectivo vector gradiente aponta no sentido do lado admissível da restrição. Contudo, é possível que $\nabla g_j(\mathbf{x})$ (ou $\nabla h_i(\mathbf{x})$) possam ser nulos devido à representação algébrica de $g_j(\mathbf{x})$ (ou $h_i(\mathbf{x})$). Por este motivo, são introduzidas condições de regularidade por forma a que nenhum dos gradientes das restrições possa ser nulo. A regularidade de um ponto é definida da seguinte forma:

Definição 2.1.6. (**Ponto Regular**) *Dado um ponto* $\mathbf{x}^*$ *e o conjunto de restrições activas* $R(\mathbf{x}^*)$*, diz-se que o ponto* $\mathbf{x}^*$ *é regular se o conjunto dos gradientes das restrições activas* $\{\nabla g_j(\mathbf{x}), j \in R(\mathbf{x}^*)\} \cup \{\nabla h_i(\mathbf{x}), i \in R(\mathbf{x}^*)\}$*, é linearmente independente.*

As condições de optimalidade, que são apresentadas em seguida, assumem sempre que uma solução $\mathbf{x}^*$ do problema de optimização é um ponto regular. As condições necessárias de primeira ordem podem ser enunciadas da seguinte forma.

Teorema 2.1.4. *(**Condições Necessárias de Primeira Ordem**) Seja* $\mathbf{x}^*$ *um mínimo local. Se* $\mathbf{x}^*$ *é um ponto regular das restrições então existe um vector de multiplicadores de Lagrange* λ^* *tal que as seguintes condições são satisfeitas em* $(\mathbf{x}^*, \lambda^*)$

$$\nabla_x L(\mathbf{x}^*, \lambda^*) = 0 \tag{2.1.4}$$

$$g_j(\mathbf{x}^*) \geq 0 \text{ para todo o } j = 1, \ldots, m \tag{2.1.5}$$

$$h_i(\mathbf{x}^*) = 0 \text{ para todo o } i = m+1, \ldots, m+p \tag{2.1.6}$$

$$\lambda_j^* > 0 \text{ para todo o } j = 1, \ldots, m \tag{2.1.7}$$

$$\lambda_j^* g_j(\mathbf{x}^*) = 0 \text{ para todo o } j = 1, \ldots, m \tag{2.1.8}$$

$$\lambda_j^* h_j(\mathbf{x}^*) = 0 \text{ para todo o } i = m+1, \ldots, m+p \tag{2.1.9}$$

Estas condições necessárias de primeira ordem são também conhecidas por *condições de Karush-Kuhn-Tucker.*

As condições necessárias de segunda ordem envolvem a matriz das segundas derivadas da função lagrangeana $\nabla^2 L(\mathbf{x}, \lambda)$.

Teorema 2.1.5. *(**Condições Necessárias de Segunda Ordem**) Seja* $\mathbf{x}^*$ *um mínimo local que é regular e* λ^* *um vector de multiplicadores de Lagrange que verifica as condições de Karush-Kuhn-Tucker, então*

$$t^T \nabla_{xx}^2 L(\mathbf{x}^*, \lambda^*) t \geq 0 \text{ para todo o } t \text{ tal que} \tag{2.1.10}$$

$$\begin{cases} \nabla g_j(\mathbf{x}^*)^T t = 0 \text{ para todo o } j = 1, \ldots, m \text{ e } j \in R(\mathbf{x}^*) \text{ com } \lambda^* > 0 \\ \nabla g_j(\mathbf{x}^*)^T t \geq 0 \text{ para todo o } j = 1, \ldots, m \text{ e } j \in R(\mathbf{x}^*) \text{ com } \lambda^* = 0 \\ \nabla h_i(\mathbf{x}^*)^T t = 0 \text{ para todo o } i = m+1, \ldots, m+p \end{cases} . \quad (2.1.11)$$

As condições suficientes de segunda ordem não exigem que $\mathbf{x}^*$ seja ponto regular.

Teorema 2.1.6. *(**Condições Suficientes de Segunda Ordem**) Seja $\mathbf{x}^*$ um ponto admissível para o qual existe um vector de multiplicadores de Lagrange λ^* que verifica as condições de Karush-Kuhn-Tucker, se*

$$t^T \nabla^2_{xx} L(\mathbf{x}^*, \lambda^*) t > 0 \text{ para todo o } t \text{ que verifica as condições (2.1.11) e } t \neq 0 \quad (2.1.12)$$

então $\mathbf{x}^*$ é um mínimo local forte.

2.1.5 Complexidade dos Problemas

A classificação da complexidade computacional dos problemas tem sido alvo de investigação e faz parte da chamada Teoria da Complexidade [GJ79]. A distinção entre problemas "fáceis" ou tratáveis e problemas "difíceis" ou intratáveis reveste-se de grande importância dada a natureza de um grande número de problemas de engenharia. Neste contexto, um *problema* pode ser caracterizado pela:

- descrição de todos os seus *parâmetros*;
- definição das propriedades que a *solução* deve satisfazer.

Uma *instância* de um problema é obtida pela especificação de valores particulares para todos os parâmetros do problema. Os *algoritmos* são procedimentos para resolver problemas. Em geral, existe interesse em encontrar o algoritmo mais "eficiente" para resolver um problema. A noção de eficiência envolve todos os recursos necessários à execução do algoritmo. Em geral, entende-se pelo algoritmo mais "eficiente" aquele que é mais rápido. Os

requisitos de tempo de um algoritmo são expressos em termos do *tamanho* de uma instância de um problema, que pretende reflectir a quantidade de dados de entrada necessários para descrever essa instância. A *função complexidade de tempo* para um algoritmo expressa os seus requisitos de tempo dando, para cada possível quantidade de dados de entrada, a maior quantidade de tempo necessária pelo algoritmo para resolver uma instância do problema desse tamanho.

Uma função $a(n)$ é $\mathcal{O}(b(n))$ quando existe uma constante c tal que $|a(n)| \leq c|b(n)|$ para todo o $n \geq 0$. Um algoritmo de *tempo polinomial* é um cuja função complexidade de tempo é $\mathcal{O}(p(n))$ para algum polinómio $p(n)$ e onde n é a quantidade de dados de entrada. Qualquer algoritmo cuja função complexidade de tempo não é limitada pela anterior é chamado de algoritmo de *tempo exponencial* (esta definição inclui certas funções complexidade de tempo não polinomiais e que não são normalmente encaradas como funções exponenciais). Esta distinção entre algoritmos tem particular importância quando se procura a solução de instâncias de problemas de grande dimensão. O tempo de execução de um algoritmo com uma função complexidade exponencial cresce muito mais à medida que o tamanho da instância do problema aumenta, do que um com uma função complexidade polinomial. Por este motivo, os algoritmos com função complexidade de tempo polinomial são preferíveis aos com função complexidade de tempo exponencial.

Um problema diz-se ter classe de complexidade polinomial (Classe **P**) se existir um algoritmo determinístico para o resolver que seja polinomial, i.e., que tenha função complexidade de tempo polinomial. Estes problemas são, em geral, tratáveis. Os problemas que não pertencem a esta classe são referidos como pertencendo à Classe **não-P**. Para estes problemas não existem algoritmos determinísticos que os resolvam em tempo polinomial. Uma outra classe de problemas é a classe não determinística polinomial (Classe **NP**). Para um problema desta classe (em geral, designado **NP**-difícil) embora não exista necessariamente um algoritmo determinístico que o resolva em tempo polinomial, a ve-

rificação de uma solução pode ser feita em tempo polinomial. Qualquer problema da classe **P** pertence necessariamente à classe **NP**. A classe **NP** é, em geral, caracterizada em termos de algoritmos não determinísticos. Estes algoritmos podem ser vistos como consistindo em duas fases: uma fase de "adivinhação" de soluções (*guessing*) seguida de outra de verificação das soluções. Para um problema da classe **NP** somente um algoritmo não determinístico o poderá resolver em tempo polinomial. Uma sub-classe importante dos problemas **NP**-difíceis são os chamados problemas **NP**-completos. Qualquer algoritmo que resolva um problema **NP**-completo pode ser traduzido para resolver um qualquer problema **NP**-difícil. Se qualquer problema **NP** é intratável, então assim o são todos os problemas **NP**-completos. Existe um conjunto de problemas que se demonstrou serem problemas **NP**-completos [GJ79]. Em geral, os problemas de PIMNL pertencem à classe **NP** pelo que não são conhecidos algoritmos determinísticos capazes de os resolver em tempo polinomial.

2.2 Computação Evolucionária

O termo Computação Evolucionária (CE) é utilizado em geral para referir uma classe de técnicas computacionais inspiradas nos princípios da evolução natural. Os primeiros Algoritmos Evolucionários (AEs) surgiram nos finais da década de 60 do século XX. Na Alemanha, foram desenvolvidas por Rechenberg [Rec73] as Estratégias Evolutivas (EEs). Na mesma época, nos Estados Unidos da América, surgiram os Algoritmos Genéticos (AGs) concebidos por Holland [Hol75]. Cada uma destas abordagens surgiu de forma independente e apresenta características muito próprias. No entanto, ambas são inspiradas nos mesmos princípios da evolução natural.

As EEs foram desenvolvidas com o objectivo de serem aplicadas a problemas de engenharia. As EEs revelaram-se algoritmos de optimização robustos e eficientes, não exigindo, ao contrário de outros algoritmos de optimização, nenhuma condição relativa à continui-

dade e convexidade do espaço de procura [Sch95].

Os AGs surgiram como algoritmos de procura aplicados a problemas de optimização. No entanto, os AGs foram desenvolvidos tendo como objectivos: o bom desempenho em optimização global e, por outro lado, a robustez no sentido de serem aplicáveis a uma grande variedade de problemas de optimização [Gol89].

Para além destas duas abordagens evolucionárias, na mesma época, surgiu, nos Estados Unidos da América, a Programação Evolucionária (PE) desenvolvida por Fogel [FOW66]. A PE foi originalmente desenvolvida no contexto da Inteligência Artificial e aplicada a espaços de procura discretos. No entanto, a sua aplicação a problemas de optimização com variáveis reais só surgiu mais tarde. Esta recente extensão da PE partilha algumas das características das EEs, apesar das aplicações em optimização serem em menor número. Por este motivo, a PE não será alvo de estudo aprofundado nesta dissertação.

A Programação Genética (PG) consiste na utilização de AGs para desenvolver programas eficientes na resolução de uma dada tarefa [Koz92]. Logo, na PG, em vez de variáveis de decisão, um programa representa um procedimento para resolver uma dada tarefa. No entanto, no contexto específico da PG, diversos mecanismos avançados foram desenvolvidos com vista a aumentar o desempenho destes algoritmos na obtenção de programas eficientes.

Mais recentemente, novos algoritmos inspirados na natureza foram propostos, tais como, o algoritmo das Colónias de Formigas de Dorigo, Maniezzo e Colorni [DMC96] e o *Particle Swarm Algorithm* de Kennedy e Eberhart [KE95].

Cada uma destas abordagens surgiu de forma independente e apresenta características muito próprias. No entanto, todas elas são inspiradas nos mesmos princípios da evolução natural e as diferentes técnicas que implementam, no seu todo, constituem a Computação Evolucionária.

2.2.1 Componentes Básicos dos Algoritmos Evolucionários

Apesar da grande diversidade de AEs existentes, podem ser identificados diversos aspectos que são comuns a todos eles. Em geral, nos AEs podem ser encontrados os três seguintes componentes básicos (quando aplicados à resolução de problemas de optimização):

- conjunto de indivíduos que representam potenciais soluções mantidas numa *população*;
- mecanismo de *selecção* enfatizando a sobrevivência dos indivíduos que representam soluções de maior qualidade para o problema;
- mecanismos de exploração do espaço de procura (e controlo da diversidade), em geral, os *operadores genéticos*.

As potenciais soluções para o problema são chamadas de *indivíduos*. Um conjunto das potenciais soluções é mantido numa *população*. As soluções de problemas de optimização como os formulados em (2.1.1) são vectores de variáveis de decisão $\mathbf{x}$. Os AEs utilizam, em geral, estruturas que representam essas soluções, e.g., um vector de valores reais, uma sequência de dígitos binários, ou mesmo, estruturas mais complexas. Logo, os AEs fazem a procura neste espaço de representações das soluções, i.e., o *espaço dos indivíduos* Φ. Cada indivíduo ϕ da população pertence ao espaço dos indivíduos, i.e., $\phi \in \Phi$. A descodificação de um indivíduo ϕ para o correspondente vector de variáveis de decisão $\mathbf{x}$ pode ser expressa pela função de descodificação $\mathbf{x} = \mathbf{d}(\phi)$.

A *selecção* faz com que os indivíduos com melhor desempenho (em termos do problema que está a ser resolvido) tendam a reproduzir-se e, consequentemente, tendam a manter-se presentes na população. Por outro lado, os indivíduos com pior desempenho tendem a ser eliminados. Este mecanismo implica a existência de uma medida do desempenho que exprima a qualidade dos indivíduos para o problema que está a ser resolvido. Por

este motivo, o desempenho dos indivíduos é medido utilizando uma *função de medida do desempenho* $F(\mathbf{x})$. Para problemas de optimização, esta função de medida do desempenho $F(\mathbf{x})$ está relacionada com a função objectivo $f(\mathbf{x})$ e com as restrições (a função de medida do desempenho constitui a ligação do AE com o problema que está a ser resolvido). Esta função de medida do desempenho deve ser tal que valorize os indivíduos que representam soluções de maior qualidade para o problema. De notar que para medir o desempenho de um indivíduo ϕ, os seguintes passos devem ser considerados:

1. descodificação do indivíduo ϕ para o correspondente vector de variáveis de decisão $\mathbf{x}$;

2. cálculo dos valores da função objectivo $f(\mathbf{x})$ e das restrições (caso existam) correspondentes ao vector de variáveis de decisão $\mathbf{x}$;

3. cálculo do valor da função de medida do desempenho $F(\mathbf{x})$ (com base nos valores calculados no passo anterior).

Nos AEs, a exploração de novas regiões do espaço de procura é feita através de *operadores genéticos*. Os operadores genéticos proporcionam a diversidade na população necessária à procura eficiente de soluções para o problema. Em geral, dois tipos principais de operadores genéticos são considerados: a *recombinação* (ou *cruzamento*) e a *mutação*. Estes operadores imitam mecanismos que ocorrem nos sistemas biológicos. Qualquer operador genético actua ao nível dos indivíduos presentes na população, i.e., no espaço dos indivíduos. A sua aplicação permite gerar novos indivíduos representando novas potenciais soluções para o problema que está ser resolvido.

Os AEs são algoritmos iterativos que iniciam a procura a partir de uma população inicial de indivíduos que, em geral, são gerados aleatoriamente. Cada iteração é, por analogia com os sistemas biológicos, chamada de *geração*. Em cada geração, os mecanismos de selecção e os operadores genéticos actuam sobre a população de indivíduos. São considerados critérios

de paragem que definem as condições que se devem verificar para terminar o processo iterativo (e.g., a obtenção de uma solução de qualidade suficiente para a resolução do problema).

2.2.2 Algoritmos Evolucionários versus Algoritmos Tradicionais

A descrição genérica dos AEs atrás apresentada torna evidente a existência de diferenças importantes relativamente aos algoritmos de optimização tradicionais. Por este motivo, em seguida, as características genéricas dos algoritmos ditos tradicionais são apresentadas por forma a ser possível a sua comparação com as abordagens evolucionárias.

Tal como os AEs, os algoritmos de optimização tradicionais são iterativos. Em geral, iniciam a procura a partir de uma aproximação inicial ao óptimo que se pretende encontrar. A partir da aproximação inicial são geradas, sucessivamente, novas estimativas do óptimo. O processo iterativo é repetido até que os critérios de paragem sejam verificados. Os diversos algoritmos de optimização distinguem-se pela forma como são calculadas as novas estimativas ao longo da procura. Quase todas as abordagens utilizam os valores da função objectivo, das restrições e, muitas vezes, das primeiras e segundas derivadas destas funções. Alguns algoritmos, para calcular uma nova aproximação ao óptimo, utilizam apenas a informação relativa à aproximação actual, enquanto que outros consideram, também, a informação recolhida durante as iterações passadas.

Os algoritmos de optimização tradicionais podem ser caracterizados, em grande parte, pela descrição anterior. No entanto, convém separar os algoritmos tradicionais em dois grupos principais: os métodos directos e os métodos baseados em gradientes. Os métodos directos guiam a procura com base em informação relacionada com a função objectivo e/ou restrições. Enquanto que os métodos baseados em gradientes, para além desta informação, necessitam, também, das primeiras e/ou segundas derivadas. Por este motivo, em geral, os métodos baseados em gradientes não são eficientes quando, num problema, não se veri-

ficam as condições de diferenciabilidade e/ou continuidade das funções. De salientar que, nestes métodos, os resultados obtidos dependem grandemente das aproximações ao óptimo consideradas no início do processo iterativo e não são eficientes na resolução de problemas com espaços de procura de natureza discreta.

Em contraste com estas abordagens ditas tradicionais, os AEs:

- iniciam a procura a partir de uma população de potenciais soluções geradas de forma aleatória;
- trabalham, ao longo das gerações, com populações de potenciais soluções, em vez de uma única aproximação ao óptimo por iteração;
- não utilizam nenhuma informação relativa às primeiras e/ou segundas derivadas da função objectivo e/ou restrições;
- podem utilizar mecanismos que permitem encontrar múltiplos óptimos locais (se existirem) numa única execução;
- guiam a procura com base em mecanismos e/ou regras probabilísticas.

Estas características dos AEs fazem com que sejam particularmente eficientes na resolução de muitos problemas reais, onde o espaço de procura muitas vezes contém um grande número de óptimos locais e/ou mais do que uma solução óptima. Por outro lado, muitos problemas reais apresentam espaços de procura de natureza discreta onde não é possível garantir as condições de diferenciabilidade e continuidade desejáveis para muitos dos algoritmos tradicionais.

No entanto, apesar das especificidades de cada algoritmo, em geral, é desejável que estes possuam as seguintes propriedades:

- *robustez*, i.e., devem apresentar um bom desempenho numa larga variedade de problemas;

- *eficiência*, i.e., não devem requerer muito tempo de computação ou espaço de memória;

- *precisão*, i.e., devem encontrar uma boa aproximação ao óptimo.

Capítulo 3

Algoritmos Evolucionários

O termo Algoritmo Evolucionário (AE) é utilizado, genericamente, na descrição de sistemas computacionais para a resolução de problemas que utilizam como elementos chave na sua implementação, modelos computacionais baseados em mecanismos de evolução. De referir que estes mecanismos de evolução correspondem ou relacionam-se com processos evolutivos biológicos. Tal como já foi atrás exposto, os AEs são, em muitos aspectos, diferentes dos métodos de procura e de optimização tradicionais.

Um grande número de aplicações de AEs a problemas de optimização uni-objectivo encontram-se descritas na literatura. No entanto, devido às suas características, os AEs mais utilizados em optimização são as Estratégias Evolutivas (EEs) e os Algoritmos Genéticos (AGs) e, por esse motivo, serão focados com detalhe nas Secções 3.1 e 3.2. Estas abordagens serão apresentadas como algoritmos para a resolução de problemas de Programação Não Linear (PNL) como os formulados em (2.1.1).

3.1 Estratégias Evolutivas

As EEs foram desenvolvidas, inicialmente, por Rechenberg [Rec73], com o objectivo de resolver problemas de optimização, tendo como base as estruturas e os processos de optimização que ocorrem na natureza. Mais tarde, Schwefel [Sch81] desenvolveu novos esque-

mas evolutivos com base nos mesmos princípios.

Nas EEs é utilizada uma população com um determinado número de indivíduos. Em cada geração, a partir dos indivíduos presentes na população, os *progenitores*, são gerados novos indivíduos, os *descendentes*. Desta forma, ao longo das gerações, novas populações são sucessivamente geradas.

As EEs trabalham directamente com a representação real das variáveis de decisão, pelo que cada indivíduo é um vector de números reais (as variáveis de decisão) representando uma potencial solução para o problema de optimização. A função de descodificação que faz o mapeamento entre um indivíduo ϕ e o correspondente vector das variáveis de decisão é $\mathbf{d}(\phi) = \mathbf{x}$ (uma vez que, $\phi = \mathbf{x}$). Como já foi referido, as EEs requerem apenas informação da função objectivo e restrições, não necessitando de derivadas ou qualquer outro conhecimento adicional.

Desde o seu aparecimento, tem vindo a ser utilizada uma nomenclatura própria para designar as diferentes EEs. Esta nomenclatura é baseada no número de progenitores, no número de descendentes e no tipo de selecção considerado. O número de progenitores é designado por μ e o número de descendentes por λ. Por outro lado, dois tipos de selecção foram originalmente descritos e designados por selecção '+' e selecção ','. A primeira e mais simples EE, desenvolvida por Rechenberg, onde a selecção era feita sobre uma população de dois membros, i.e., $\mu + \lambda = 1 + 1 = 2$, é designada, na nomenclatura atrás apresentada, por EE-(1+1). Posteriormente, o mesmo autor desenvolveu uma estratégia multi-membros mais complexa, onde a selecção era feita sobre uma população de $\mu > 1$ indivíduos e um descendente; i.e., $\mu + \lambda = \mu + 1$, que é designada por EE-$(\mu + 1)$.

De uma forma mais genérica, numa EE-$(\mu + \lambda)$, numa determinada geração, existe uma população de μ progenitores que gera λ descendentes por mutação (Figura 3.1). Em seguida, os $\mu + \lambda$ indivíduos são avaliados e ordenados de acordo com os seus valores da função objectivo (a medida do desempenho considerada é a própria função objectivo, i.e.,

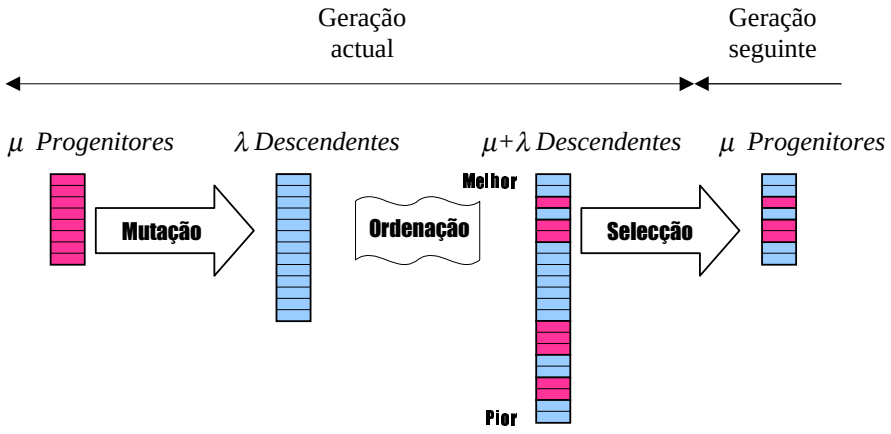

Figura 3.1: Aspecto Geral da Estratégia Evolutiva $(\mu + \lambda)$

$F(\mathbf{x}) = f(\mathbf{x})$). Finalmente, o processo de selecção faz com que apenas os μ melhores de todos os $\mu + \lambda$ indivíduos se tornem os progenitores da geração seguinte, i.e., a selecção é feita a partir dos $\mu + \lambda$ indivíduos. De referir que este mecanismo de selecção é determinístico, uma vez que apenas os melhores indivíduos são seleccionados para formarem a população de progenitores da geração seguinte.

Um outro modelo conceptual pode ser definido onde, numa determinada geração, uma população de μ progenitores gera, por mutação, λ descendentes (assumindo que $\lambda > \mu$). Em seguida, os λ descendentes são avaliados e ordenados de acordo com os seus valores da função objectivo. Então, os μ melhores dos λ descendentes gerados tornam-se os progenitores da próxima geração, i.e., os μ progenitores não são incluídos no processo de selecção. Esta EE que utiliza a selecção ',' é designada por EE-(μ, λ) (Figura 3.2). Os dois tipos de EEs atrás descritas, a EE-$(\mu + \lambda)$ e a EE-(μ, λ), diferem basicamente no procedimento de selecção.

Originalmente, as EEs baseavam-se num único operador, a mutação, para gerar novos indivíduos. Posteriormente, foi introduzido um outro operador, a recombinação, que era aplicado juntamente com a mutação [Sch95]. Uma vez que, originalmente, as EEs evolutivas

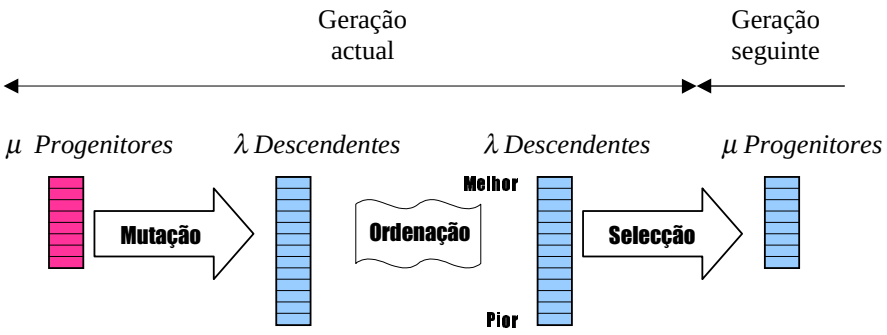

Figura 3.2: Aspecto Geral da Estratégia Evolutiva (μ, λ)

faziam a procura de novos pontos exclusivamente utilizando a mutação, optou-se, nesta fase, por apresentar as diferentes EEs sem recombinação. Mais adiante, nesta mesma secção, descrever-se-á detalhadamente este operador.

Em seguida, cada uma das EEs atrás focadas irá ser descrita com maior detalhe. Todos os algoritmos apresentados são formulados para problemas de minimização. Nesta fase, apenas se considerarão restrições do tipo desigualdade. Mais adiante, ir-se-á focar o problema do tratamento de restrições do tipo igualdade.

3.1.1 Estratégia Evolutiva $(1 + 1)$

Este esquema evolutivo com apenas 2 membros, designado por EE-$(1 + 1)$, foi proposto por Rechenberg [Rec64] para a resolução de problemas de optimização. Numa determinada geração, existe apenas um progenitor $(\mu = 1)$ e um único descendente $(\lambda = 1)$, e a selecção tem lugar apenas entre estes dois membros.

O Algoritmo 3.1.1 é o algoritmo da EE-$(1 + 1)$. A procura inicia-se a partir de um ponto inicial $\mathbf{x}^{(0)}$ (uma aproximação ao óptimo). Em seguida, um novo ponto $\mathbf{x}_N^{(k)}$ é gerado por mutação através da adição de uma quantidade aleatória $\mathbf{z}^{(k)}$. Depois, os dois pontos são comparados e o melhor (com menor valor da função objectivo e satisfazendo todas as

restrições) é seleccionado para progenitor da próxima geração. Este processo é repetido até que o critério de paragem seja verificado.

Algoritmo 3.1.1. - *ESTRATÉGIA EVOLUTIVA* $(1+1)$

1. INICIAÇÃO

$$\mathbf{x}^{(0)} = \begin{bmatrix} x_1^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix} \textit{ tal que } g_j(\mathbf{x}^{(0)}) \geq 0 \textit{ para todo o } j = 1, \ldots, m$$

$k = 0$

2. MUTAÇÃO

$$\mathbf{x}_N^{(k)} = \mathbf{x}^{(k)} + \mathbf{z}^{(k)} = \begin{bmatrix} x_1^{(k)} + z_1^{(k)} \\ \vdots \\ x_n^{(k)} + z_n^{(k)} \end{bmatrix}$$

3. SELECÇÃO

$$\mathbf{x}^{(k+1)} = \begin{cases} \mathbf{x}_N^{(k)} \leftarrow f(\mathbf{x}_N^{(k)}) \leq f(\mathbf{x}^{(k)}) \wedge g_j(\mathbf{x}_N^{(k)}) \geq 0 \textit{ para todo o } j = 1, \ldots, m \\ \mathbf{x}^{(k)} \leftarrow \textit{caso contrário} \end{cases}$$

$k = k + 1$

SE não se verificar o critério de paragem ENTÃO voltar ao passo 2. SENÃO terminar.

Mutação

O operador de mutação consiste em gerar um novo ponto $\mathbf{x}_N^{(k)}$ através da adição de uma quantidade aleatória $\mathbf{z}^{(k)}$. A quantidade aleatória $\mathbf{z}^{(k)}$ traduz o tamanho do passo (ou comprimento do deslocamento) na geração ou iteração k, e a sua escolha deve ser feita de tal forma que pequenos deslocamentos ocorram frequentemente e grandes deslocamentos ocorram raramente. Com este propósito, em geral, as quantidades aleatórias $\mathbf{z}^{(k)}$ são geradas de acordo com uma distribuição Normal. Para além disso, as seguintes condições devem ser impostas à distribuição dos tamanhos do passo:

1. O valor esperado dos componentes $z_i^{(k)}$ de $\mathbf{z}^{(k)}$, com $i = 1, \ldots, n$, deve ser nulo, i.e., $E[z_i^{(k)}] = 0$;

2. As variâncias σ_i^2, com $i = 1, \ldots, n$, devem ter valores pequenos.

Logo, os componentes aleatórios $z_i^{(k)}$ podem ser calculados de acordo com uma distribuição Normal com média nula e variância σ_i^2, i.e., $z_i^{(k)} \sim N(0, \sigma_i^2)$ (mais detalhes acerca da geração de números aleatórios normalmente distribuídos podem ser encontrados no Apêndice B) .

Aproximação Inicial

Como já foi dito, para iniciar a procura é necessária uma aproximação inicial ao óptimo $\mathbf{x}^{(0)}$. Mas, para além disso, é necessário escolher os valores iniciais para os desvios padrão σ_i. Os valores iniciais típicos para os desvios padrão σ_i podem ser expressos pela equação (3.1.1) onde Δx é uma medida aproximada da distância esperada da aproximação inicial $\mathbf{x}^{(0)}$ ao óptimo e n é a dimensão do problema (o número de variáveis de decisão),

$$\sigma_i^{(0)} = \frac{\Delta x}{\sqrt{n}}. \tag{3.1.1}$$

Ao longo da procura, deve-se garantir as seguintes duas condições para os valores dos tamanhos do passo σ_i:

1. $\sigma_i > 0$ para $i = 1, \ldots, n$;
2. os valores de σ_i devem ser suficientemente grandes para que pelo menos o dígito menos significativo da variável x_i seja alterado.

As condições anteriores podem ser escritas em termos dos seguintes limites inferiores para os tamanhos do passo: $\sigma_i^{(k)} \geq \varepsilon_1$ e $\sigma_i^{(k)} \geq \varepsilon_2 \left| x_i^{(k)} \right|$ para $i = 1, \ldots, n$, onde ε_1 e ε_2 dependem da precisão do computador utilizado, sendo $\varepsilon_1 > 0$ e $\varepsilon_2 > 0$.

Controlo do Tamanho do Passo

Para que o processo de optimização seja eficiente, os deslocamentos devem ser continuamente modificados. Se os deslocamentos forem demasiado pequenos, o número de iterações

do processo de procura é desnecessariamente grande; pelo contrário, se forem demasiado grandes, poderá não se obter uma boa aproximação ao óptimo, ou mesmo, o processo poderá não convergir. Por isso, em todas as estratégias de optimização, o controlo do tamanho do passo é uma das componentes mais importantes no processo de procura.

Nas EEs, a grandeza das quantidades aleatórias $z_i^{(k)}$ depende dos desvios padrão $\sigma_i^{(k)}$. Quanto maiores forem os desvios padrão $\sigma_i^{(k)}$, maiores serão as quantidades aleatórias $z_i^{(k)}$ e, consequentemente, os tamanhos do passo na geração k. Rechenberg [Rec73], baseando-se na aplicação da EE-$(1+1)$ a alguns tipos de problemas de optimização, formulou a chamada "Regra de 1/5 de Sucessos" para controlar os tamanhos do passo. Esta regra pode ser escrita da seguinte forma:

> "De tempos a tempos, ao longo do processo de procura, calcule-se a frequência de sucessos, i.e., a razão entre o número de sucessos e o número de iterações. Se a razão for maior que 1/5, aumente-se a variância, se for menor que 1/5, diminua-se a variância."

Assumindo-se que esta regra é aplicada periodicamente todas as t gerações, a sua expressão pode ser definida da seguinte forma:

$$\sigma_i^{(k+1)} = \begin{cases} c_{dec}\sigma_i^{(k)} \leftarrow \Pr_{suc}(t) < 1/5 \\ c_{inc}\sigma_i^{(k)} \leftarrow \Pr_{suc}(t) > 1/5 \\ \sigma_i^{(k)} \leftarrow \Pr_{suc}(t) = 1/5 \end{cases}, \tag{3.1.2}$$

onde $\Pr_{suc}(t)$ é a proporção de sucessos nas últimas t gerações e, $c_{dec} < 1$ e $c_{inc} > 1$ são, respectivamente, os coeficientes de diminuição e de aumento dos desvios padrão σ_i.

Defina-se a razão de progresso como o valor esperado da diferença radial após uma iteração (Figura 3.3):

$$\varphi^{(k)} = E\left[\left\|\mathbf{x}^* - \mathbf{x}^{(k)}\right\| - \left\|\mathbf{x}^* - \mathbf{x}^{(k+1)}\right\|\right] = E\left[r^{(k)} - r^{(k+1)}\right], \tag{3.1.3}$$

onde $\mathbf{x}^*$ representa o óptimo e, $\mathbf{x}^{(k)}$ e $r^{(k)}$ representam, respectivamente, a aproximação ao óptimo e a distância ao óptimo na iteração k.

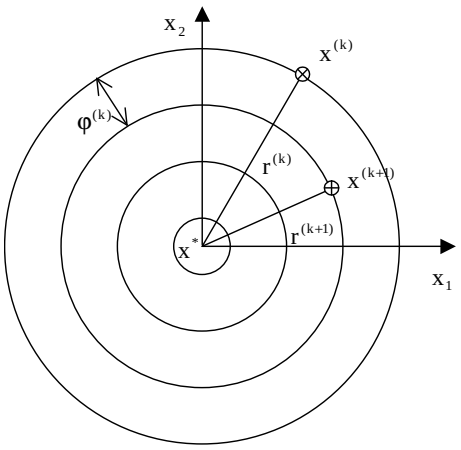

Figura 3.3: Razão de Progresso

Verifica-se que, para muitos problemas, a "Regra de 1/5 de Sucessos" mostra-se extremamente eficiente na manutenção, aproximadamente, da maior razão de progresso possível para o óptimo. No entanto, interessa definir qual a frequência com que o critério de sucesso deve ser testado (o valor do parâmetro t) e qual o factor de diminuição ou de aumento dos desvios padrão mais eficiente (os valores dos coeficientes c_{dec} e c_{inc}). Para tentar responder a esta questão, podem-se utilizar os resultados obtidos por Rechenberg [Rec73]. A máxima razão de progresso é dada por:

$$\varphi_{\max} = c_1 \frac{r^{(k)}}{n} \text{ sendo } c_1 \cong 0.2025, \tag{3.1.4}$$

com variância comum σ^2, cujo valor σ^2_{opt} óptimo é dado por:

$$\sigma^2_{opt} = \left(c_2 \frac{r^{(k)}}{n} \right)^2 \text{ com } c_2 \cong 1.224, \tag{3.1.5}$$

para todos os componentes $z_i^{(k)}$ do vector aleatório $\mathbf{z}^{(k)}$. Nestas expressões, $r^{(k)}$ é a distância actual ao óptimo e n é o número de variáveis. Destas equações pode-se obter a relação para as variações nos comprimentos dos deslocamentos após uma geração, supondo a condição

de máxima razão de convergência:

$$\frac{\sigma_{opt}^{(k+1)}}{\sigma_{opt}^{(k)}} = \frac{r^{(k+1)}}{r^{(k)}} = \frac{r^{(k)} - \varphi_{\max}}{r^{(k)}} = 1 - \frac{c_1}{n}, \tag{3.1.6}$$

ou após n gerações:

$$\frac{\sigma_{opt}^{(k+n)}}{\sigma_{opt}^{(k)}} = \left(1 - \frac{c_1}{n}\right)^n. \tag{3.1.7}$$

Quando n for muito maior que 1, o factor do comprimento do deslocamento tende para uma constante:

$$\lim_{n \to +\infty} \left(1 - \frac{c_1}{n}\right)^n = e^{-c_1} \cong 0.817 \cong \frac{1}{1.224}. \tag{3.1.8}$$

Este resultado aplica-se ao caso limite em que existem muitas variáveis (n grande) e está de acordo com a equação (3.1.4) que expressa que a razão de progresso é inversamente proporcional ao número de variáveis. O facto da razão de progresso perto do seu máximo ser bastante insensível a pequenas variações das variâncias, juntamente com o facto da probabilidade de sucesso só poder ser determinada pela média de diversas gerações, conduz à seguinte formulação mais precisa da "Regra de 1/5 de Sucessos" onde n é o número de variáveis de decisão do problema de optimização:

> "Após cada n gerações, verifique-se quantos sucessos ocorreram nas últimas $10n$ gerações. Se este número for inferior a $2n$, multipliquem-se os comprimentos dos deslocamentos pelo factor 0.85; senão, dividam-se os comprimentos dos deslocamentos pelo factor 0.85."

Esta regra permite que os comprimentos dos deslocamentos ou as variâncias dos deslocamentos aleatórios sejam controlados, fixando-se os seguintes valores $c_{dec} = 0.85$, $c_{inc} = 1/0.85$ e $t = 10n$. No entanto, a probabilidade de sucesso não fornece nenhuma indicação de quão apropriadas são as razões das variâncias σ_i^2 relativamente umas às outras (para cada variável), pelo que os comprimentos dos deslocamentos só podem ser todos reduzidos em conjunto ou todos aumentados em conjunto. Por este motivo, em geral, considera-se que $\sigma_1^2 = \sigma_2^2 = \ldots = \sigma_n^2 = \sigma^2$.

Critério de Paragem

O critério de paragem determina quando o processo de procura deve ser terminado. Em geral, são utilizadas as condições:

$$\left|f(\mathbf{x}_N^{(k-\Delta k)}) - f(\mathbf{x}_N^{(k)})\right| \leq \varepsilon_3 \text{ ou } \frac{\left|f(\mathbf{x}_N^{(k-\Delta k)}) - f(\mathbf{x}_N^{(k)})\right|}{\left|f(\mathbf{x}_N^{(k)})\right|} \leq \varepsilon_4, \tag{3.1.9}$$

e $\Delta k \geq 20n$, onde ε_3 e ε_4 dependem da precisão do computador utilizado, sendo $\varepsilon_3 > 0$ e $\varepsilon_4 > 0$. A condição $\Delta k \geq 20n$ assegura que, no caso extremo, entre testes da "Regra de 1/5 de Sucessos", os comprimentos dos deslocamentos são reduzidos ou aumentados, pelo menos pelo factor $(0.85)^{\pm 20} \cong (25)^{\mp 1}$, o que evita que a procura seja terminada apenas porque os comprimentos dos deslocamentos são forçados a variar muito frequentemente. Por outro lado, é evidente que o critério de convergência não precisa de ser verificado em todas as gerações. O procedimento usual é testá-lo apenas cada $20n$ gerações.

Tratamento de Restrições

O esquema evolutivo, tal como foi apresentado, prevê o tratamento de restrições de desigualdade do tipo $g_j(\mathbf{x}) \geq 0$ com $j = 1, \ldots, m$. Em cada geração, se a mutação gerar um ponto que não satisfaz as restrições, este não é aceite, i.e., utiliza-se um mecanismo de eliminação das soluções não admissíveis. Desta forma, a procura restringe-se ao interior da região admissível. No entanto, é muitas vezes difícil especificar um vector inicial $\mathbf{x}^{(0)}$ que seja uma aproximação ao óptimo e que satisfaça todas as restrições (se for utilizado um vector inicial que não satisfaça as restrições poderá demorar muito tempo até que o processo de procura determine um ponto da região admissível). Um dos processos para a determinação de um vector inicial na região admissível é o descrito por Box [Box65]. Neste processo, uma função objectivo auxiliar $r(\mathbf{x})$ representando a soma dos valores das funções

das restrições violadas é construída:

$$r(\mathbf{x}) = \sum_{j=1}^{m} g_j(\mathbf{x})\delta_j(\mathbf{x}), \tag{3.1.10}$$

onde

$$\delta_j(\mathbf{x}) = \begin{cases} -1 \text{ se } g_j(\mathbf{x}) < 0 \\ 0 \text{ se } g_j(\mathbf{x}) \geq 0 \end{cases}.$$

Um decréscimo no valor da função $r(\mathbf{x})$ representa uma aproximação à região admissível. Assim, quando $r(\mathbf{x}) = 0$ então $\mathbf{x}$ satisfaz todas as restrições e pode ser utilizado como vector inicial. Enquanto não é encontrado um ponto admissível, a procura é feita com base na função definida em (3.1.10).

O esquema atrás descrito permite tratar apenas restrições do tipo desigualdade. As restrições do tipo igualdade $h_i(\mathbf{x}) = 0$ podem ser reformuladas como restrições de desigualdade da seguinte forma:

$$h_i(\mathbf{x}) \geq 0 \wedge -h_i(\mathbf{x}) \geq 0.$$

Para além disso, podem ser consideradas quantidades positivas e pequenas, $\underline{\varepsilon}$ e $\overline{\varepsilon}$ de tal forma que $-\underline{\varepsilon} \leq h_i(x) \leq \overline{\varepsilon}$, i.e.:

$$h_i(\mathbf{x}) + \underline{\varepsilon} \geq 0 \wedge -h_i(\mathbf{x}) + \overline{\varepsilon} \geq 0.$$

No entanto, o tratamento de restrições do tipo igualdade pode criar dificuldades às EE, dado que encontrar novos pontos que sejam admissíveis pode traduzir-se num comportamento oscilatório.

3.1.2 Estratégias Evolutivas (μ, λ) e $(\mu + \lambda)$

Nestas EEs multi-membros existem, em cada geração, μ progenitores e λ descendentes. A EE-(μ, λ) distingue-se da EE-$(\mu + \lambda)$ apenas na selecção que tem lugar, no primeiro caso, entre os λ descendentes enquanto que, no segundo caso, se faz a partir dos $\mu + \lambda$ indivíduos [Sch95].

Algoritmo 3.1.2. - *ESTRATÉGIA EVOLUTIVA* (μ, λ)

1. INICIAÇÃO

$\mathbf{x}_p^{(0)} = \begin{bmatrix} x_{p,1}^{(0)} \\ \vdots \\ x_{p,n}^{(0)} \end{bmatrix}$ *tal que* $g_j(\mathbf{x}_p^{(0)}) \geq 0$ *para todo o* $j = 1, \ldots, m$ *e* $p = 1, \ldots, \mu$

$k = 0$

2. MUTAÇÃO

$\mathbf{x}_{d,N}^{(k)} = \mathbf{x}_u^{(k)} + \mathbf{z}^{(k\lambda+d)} = \begin{bmatrix} x_{u,1}^{(k)} + z_1^{(k\lambda+d)} \\ \vdots \\ x_{u,n}^{(k)} + z_n^{(k\lambda+d)} \end{bmatrix}$ *tal que* $g_j(\mathbf{x}_{d,N}^{(k)}) \geq 0$

para todo o $j = 1, \ldots, m$, $d = 1, \ldots, \lambda$ *e* $u \in [1, \mu]$,

e.g., $u = \begin{cases} \mu \leftarrow d = \mu, 2\mu, \ldots, K\mu \text{ com } K \text{ inteiro} \\ d//\mu \leftarrow \text{caso contrário} \end{cases}$

onde // representa o resto da divisão inteira

3. SELECÇÃO

Ordenar os $\mathbf{x}_{d,N}^{(k)}$ *indivíduos, com* $d = 1, \ldots, \lambda$, *de tal forma que* $f(\mathbf{x}_{a,N}^{(k)}) \leq f(\mathbf{x}_{b,N}^{(k)})$

para todo o $a = 1, \ldots, \mu$ *e* $b = \mu, \ldots, \lambda$

$\mathbf{x}_p^{(k+1)} = \mathbf{x}_{a,N}^{(k)}$ *com* $p = 1, \ldots, \mu$ *e* $a = 1, \ldots, \mu$

$k = k + 1$

SE não se verificar o critério de paragem ENTÃO voltar ao passo 2. SENÃO terminar.

O Algoritmo 3.1.2 é o algoritmo da EE-(μ, λ). A procura é iniciada gerando μ pontos a partir de um ponto inicial (uma aproximação ao óptimo). Então, λ pontos são gerados por mutação, como resultado da adição de números aleatórios $\mathbf{z}^{(k)}$. Depois, os λ pontos gerados são ordenados de acordo com os valores da função objectivo e das restrições. Os μ melhores pontos são seleccionados para serem os progenitores na geração seguinte. Este processo é repetido até que o critério de paragem seja satisfeito.

O algoritmo da EE-$(\mu + \lambda)$ é idêntico ao anterior (Algoritmo 3.1.2), com excepção do procedimento de selecção que escolhe os melhores μ dos $\mu + \lambda$ indivíduos, para progenitores na geração seguinte.

Mutação

Tal como anteriormente, o operador de mutação consiste em gerar novos pontos pela adição de quantidades aleatórias. Nas EEs multi-membros cada progenitor produz, em média, λ/μ descendentes, de tal forma que λ descendentes sejam gerados. As mesmas condições acerca da normalidade das quantidades aleatórias $z_i^{(k)}$ são consideradas, i.e., $z_i^{(k)} \sim N(0, \sigma_i^2)$ para $i = 1, \ldots, n$.

Aproximação inicial

De forma semelhante à EE-$(1 + 1)$, a procura inicia-se a partir de uma aproximação ao óptimo $\mathbf{x}^{(0)}$, que permite gerar os μ pontos da população inicial. Para além disso, é necessário escolher os valores iniciais para os desvios padrão σ_i. Tal como na EE-$(1 + 1)$, os valores iniciais típicos para os desvios σ_i podem ser expressos pela mesma equação (3.1.1). Para além disso, as mesmas condições em termos dos limites inferiores para os comprimentos do passo devem ser consideradas, i.e., $\sigma_i^{(k)} \geq \varepsilon_1$ e $\sigma_i^{(k)} \geq \varepsilon_2 \left| x_i^{(k)} \right|$ para $i = 1, \ldots, n$, onde ε_1 e ε_2 dependem da precisão do computador utilizado, sendo $\varepsilon_1 > 0$ e $\varepsilon_2 > 0$.

Controlo do Tamanho do Passo

Os desvios padrão σ_i podem ser actualizados de diversas formas. Este processo consiste na adaptação dos parâmetros da estratégia durante a procura [Sch95]. As duas implementações mais conhecidas são:

- a Adaptação Isotrópica, e
- a Adaptação Não Isotrópica.

Em seguida, descreve-se sucintamente cada um destes esquemas de actualização dos desvios padrão.

Adaptação Isotrópica Neste esquema, tal como na EE-(1 + 1), é considerada uma variância σ^2 comum a todas as variáveis, i.e., $\sigma_1^2 = \sigma_2^2 = \ldots = \sigma_n^2$. Os desvios padrão σ_i são agora actualizados pela equação (3.1.11) onde $\mathbf{z}^{(k)}$ é determinado de acordo com uma distribuição Normal com média zero e variância $\Delta\sigma^2$, onde $\Delta\sigma$ é um parâmetro do algoritmo,

$$\sigma^{(k+1)} = \sigma^{(k)} e^{z^{(k)}} \text{ com } z^{(k)} \sim N(0, \Delta\sigma^2). \tag{3.1.11}$$

Em geral, o valor de $\Delta\sigma$ é fixado em $1/\sqrt{n}$.

Adaptação Não Isotrópica Neste esquema, os desvios padrão σ_i são actualizados (um para cada variável de decisão) de acordo com a equação:

$$\sigma_i^{(k+1)} = \sigma_i^{(k)} e^{z_i^{(k)}} e^{z^{(k)}}, \tag{3.1.12}$$

onde $z_i \sim N(0, \Delta\sigma^2)$, $\mathbf{z} \sim N(0, \Delta\sigma^{`2})$ e, $\Delta\sigma$ e $\Delta\sigma$‘ são parâmetros do algoritmo. Em geral, os valores de $\Delta\sigma$ e $\Delta\sigma$‘ são fixados, respectivamente, em $1/\sqrt{2\sqrt{n}}$ e $1/\sqrt{2n}$. Este esquema não isotrópico, ao contrário do anterior, permite adaptar os desvios padrão independentemente para cada variável de decisão.

Critério de Paragem

O critério de convergência permite definir o término do processo de procura. Em geral, o critério de convergência adoptado para a EE-$(\mu + \lambda)$ e EE-(μ, λ) é terminar a procura quando as seguintes condições são verificadas:

$$\left| f_{\max}^{(k)} - f_{\min}^{(k)} \right| \leq \varepsilon_3 \text{ ou } \frac{\left| f_{\max}^{(k)} - f_{\min}^{(k)} \right|}{\left| \overline{f}^{(k)} \right|} \leq \varepsilon_4, \tag{3.1.13}$$

onde $f_{\max}^{(k)}$, $f_{\min}^{(k)}$ e $\overline{f}^{(k)}$ são, respectivamente, o maior (pior indivíduo), o menor (melhor indivíduo) e a média dos valores da função objectivo da população de μ progenitores, ε_3 e ε_4 dependem da precisão do computador utilizado, sendo $\varepsilon_3 > 0$ e $\varepsilon_4 > 0$.

Tratamento das Restrições

As EEs multi-membros, tal como foram apresentadas, prevêem o tratamento de restrições de desigualdade do tipo $g_j(\mathbf{x}) \geq 0$ com $j = 1, \ldots, m$. Em cada geração, se a mutação gerar pontos que não satisfazem as restrições, estes não são aceites. Desta forma, a procura restringe-se ao interior da região admissível. A especificação de vector inicial $\mathbf{x}^{(0)}$ para gerar a população inicial de μ indivíduos que satisfaça todas as restrições, caso seja necessário, pode ser feita utilizando o mesmo processo descrito anteriormente e utilizado para a EE-$(1+1)$.

Tal como anteriormente foi referido para a EE-$(1+1)$, as EEs multi-membros poderão experimentar problemas no tratamento de restrições do tipo igualdade que resultam da dificuldade em gerar novos pontos que sejam admissíveis.

Recombinação

Schwefel [Sch95] notou uma aceleração considerável na procura, bem como a facilitação da adaptação dos desvios padrão pela introdução do operador de recombinação nas EEs. Basicamente, a recombinação consiste em, antes da mutação, recombinar um conjunto de progenitores por forma a encontrar uma nova solução. Seja ρ o número de progenitores que participam na recombinação ($1 \leq \rho \leq \mu$). Estes ρ progenitores são escolhidos aleatoriamente de entre os μ indivíduos da população. Quando $\rho = 1$ então não existe recombinação. Logo, a nomenclatura das EEs pode ser estendida, e as EEs com recombinação são, em geral, referidas por EE-$(\mu/\rho + \lambda)$ ou EE-$(\mu/\rho, \lambda)$. Existem dois tipos de recombinação principais:

- a Recombinação Intermédia, e
- a Recombinação Discreta.

Estes tipos de recombinação podem ser aplicados quer às variáveis de decisão quer aos desvios padrão σ_i. Em seguida, estes tipos de recombinação são descritos em termos das variáveis de decisão. A sua aplicação aos desvios padrão σ_i faz-se de forma similar.

Recombinação Intermédia Neste tipo de recombinação, os componentes dos descendentes são obtidos calculando-se a média dos componentes dos progenitores correspondentes (escolhidos aleatoriamente a partir da população). Logo, considerando ρ progenitores escolhidos aleatoriamente, os descendentes $\mathbf{x_{d,Int\_rec}}$ serão dados por:

$$\mathbf{x_{d,Int\_rec}} = \frac{1}{\rho}\sum_{m=1}^{\rho}\mathbf{x_m} = \left[\ \tfrac{1}{\rho}\textstyle\sum_{m=1}^{\rho} x_{m,1} \quad \cdots \quad \tfrac{1}{\rho}\sum_{m=1}^{\rho} x_{m,n}\ \right]^T. \tag{3.1.14}$$

Se $\rho = \mu$, este esquema tende a gerar descendentes próximos do centróide da população.

Recombinação Discreta Neste tipo de recombinação, cada componente dos descendentes é escolhido aleatoriamente a partir de um dos ρ progenitores. Logo, supondo ρ progenitores escolhidos aleatoriamente, os descendentes $\mathbf{x_{d,Dis\_rec}}$ serão dados por:

$$\mathbf{x_{d,Dis\_rec}} = \left[\ x_{u_1,1} \quad \cdots \quad x_{u_n,n}\ \right]^T \text{ com } u_1 \in U(0,\rho), \ldots, u_n \in U(0,\rho), \tag{3.1.15}$$

onde u_i, para $i = 1, \ldots, n$, são valores gerados aleatoriamente (uniformemente), indicando a partir de que progenitor (dos ρ progenitores) o valor da variável de decisão é copiado para o descendente. Este procedimento permite obter diferentes combinações dos valores das variáveis de decisão das soluções existentes na população.

3.2 Algoritmos Genéticos

Os Algoritmos Genéticos (AGs) são um modelo computacional de pesquisa probabilística, inspirados no processo de selecção natural e na genética. Os AGs foram desenvolvidos por Holland [Hol75] na Universidade de Michigan, tendo sido desde aí aplicados com sucesso

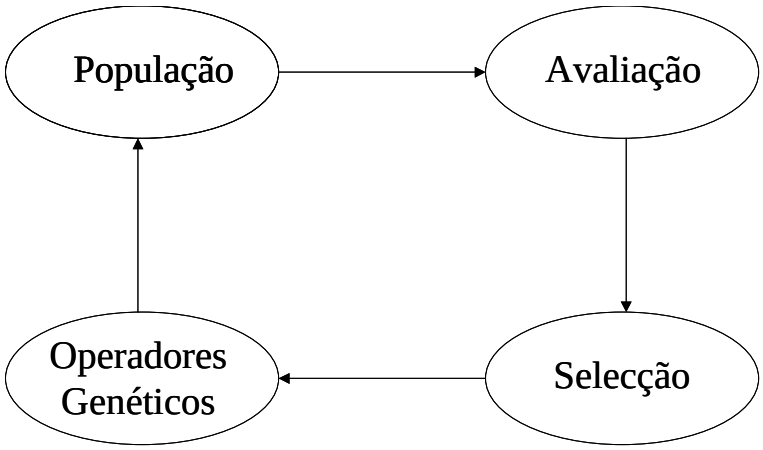

Figura 3.4: Aspecto Geral do Algoritmo Genético

em diversas áreas, nomeadamente em problemas de optimização numérica [Bet81, Bri81] ou problemas relacionados com processos adaptativos [DJ80].

Os componentes básicos de um AG são a população de indivíduos, em que cada um deles representa uma potencial solução para o problema considerado, o mecanismo de avaliação dos indivíduos da população e os operadores genéticos que pesquisam novas soluções.

A Figura 3.4 ilustra o funcionamento de um AG. Os AGs trabalham sobre uma população de potenciais soluções, ou indivíduos, às quais é aplicado o princípio da sobrevivência das melhores, i.e., os indivíduos competem entre si pela sobrevivência. Após serem avaliados, os melhores indivíduos têm uma maior probabilidade de serem escolhidos (para progenitores) e gerarem novos indivíduos (os descendentes). A geração de novos indivíduos é feita através de mecanismos baseados na genética. Assim, os descendentes são gerados a partir da recombinação dos progenitores pelo que herdam algumas das suas características. Para além disso, é aplicada mutação com objectivo de possibilitar o aparecimento de algumas características verdadeiramente inovadoras. Os descendentes gerados competem entre si e com os progenitores. Este processo é repetido ao longo de um determinado número de gerações.

Ao longo das gerações, uma vez que os melhores indivíduos têm maior probabilidade de serem seleccionados para gerar descendentes, e, possivelmente, de gerarem bons descendentes, a população tende a ter cada vez melhores indivíduos. Este processo de procura genética conduz à evolução da população e os melhores indivíduos tendem a sobreviver, tal como sucede na adaptação natural. A procura utilizando um AG, tal como foi descrito, processa-se de acordo com o Algoritmo 3.2.1.

Algoritmo 3.2.1. - ***ALGORITMO GENÉTICO***

1. INICIAÇÃO

A população inicial de indivíduos é criada aleatoriamente.

2. AVALIAÇÃO

Cada indivíduo é na população avaliado de acordo com uma medida do desempenho.

3. SELECÇÃO

Um conjunto de indivíduos é seleccionado para gerar descendentes de tal forma que os que tiverem melhor desempenho têm maior probabilidade de serem seleccionados.

4. OPERADORES GENÉTICOS

Ao conjunto de indivíduos seleccionados são aplicados operadores genéticos como a recombinação e a mutação.

SE não se verificar o critério de paragem ENTÃO voltar ao passo 2. SENÃO terminar.

3.2.1 População

Uma vez que se podem identificar diversas analogias entre os AGs e os sistemas biológicos e que, por outro lado, a nomenclatura deste sistemas é muitas vezes utilizada para referir componentes ou aspectos do funcionamento dos AGs, nesta secção começa-se por clarificar a relação entre a nomenclatura dos sistemas naturais e a dos AGs.

Tradicionalmente, nos AGs, os indivíduos são representados por sequências de dígitos binários. Em termos simples, as sequências de dígitos binários são análogas às sequências de amino-ácidos (A, C, T e G) nos cromossomas dos sistemas biológicos. Na terminologia

dos sistemas biológicos ou naturais, diz-se que os cromossomas são constituídos por genes que podem tomar um determinado número de valores chamados de alelos. Para além disso, define-se o locus do gene como sendo a sua posição no cromossoma. De forma algo análoga, nos AGs, diz-se que as sequências binárias são constituídas por dígitos binários, os *bits* (0 ou 1), que podem estar localizados em diferentes posições. Por outro lado, a posição de cada dígito binário na sequência binária define o seu significado. Cada variável de decisão x_i, com $i = 1, \ldots, n$, é representada por uma sequência de l dígitos binários $\omega_i = \langle b_{1,i} \ldots b_{l,i} \rangle_2$. Para cada variável de decisão x_i, a função de descodificação é dada por

$$\mathbf{d}(\omega) = \begin{bmatrix} d(\omega_1) \\ \vdots \\ d(\omega_n) \end{bmatrix} = \begin{bmatrix} d(\langle b_{1,1} \ldots b_{l,1} \rangle_2) \\ \vdots \\ d(\langle b_{1,n} \ldots b_{l,n} \rangle_2) \end{bmatrix}$$

onde

$$d(\omega_i) = d(\langle b_{1,i} \ldots b_{l,i} \rangle_2) = \underline{x_i} + \frac{\overline{x_i} - \underline{x_i}}{2^l - 1} \sum_{k=1}^{l} b_{k,i} 2^{l-k-1}.$$

Um dos problemas emergentes da codificação em binário padrão é que, neste código, dependendo da sua posição, a mudança de um único dígito binário pode determinar uma pequena ou grande mudança dos valores das variáveis de decisão. Para evitar que a mudança de um único dígito binário possa conduzir a uma mudança abrupta no valor da variável recorre-se ao chamado código *Gray*. Este código tem a propriedade de que valores adjacentes da variável de decisão diferem exactamente num único dígito binário numa determinada posição [Wri91] (mais detalhes sobre a codificação podem ser encontrados no Apêndice C). Bäck [Bäc96] demonstrou a superioridade da utilização, nos AGs, do código *Gray* face ao código binário padrão.

Em termos biológicos, a totalidade do pacote genético (combinação de um ou mais cromossomas que descrevem a forma de construção e operação do organismo) constitui o genótipo que corresponde, ao nível dos AGs, às chamadas estruturas. Por outro lado, a interacção do genótipo com o ambiente é definida como o fenótipo, e, analogamente, as estruturas são descodificadas para formar soluções.

Sistemas Naturais	Algoritmos Genéticos
Cromossoma	Sequência de dígitos binários
Gene	Dígito binário
Alelo	0 ou 1
Locus	Posição na sequência de dígitos binários
Genótipo	Estrutura
Fenótipo	Solução

Tabela 3.1: Terminologia dos Sistemas Naturais e dos Algoritmos Genéticos

A Tabela 3.1 resume a correspondência entre a terminologia usada nos sistemas naturais e nos AGs.

Alguns resultados teóricos parecem suportar a discretização do espaço de procura (ver Secção 3.2.9). Para além disso, a representação dos indivíduos utilizando sequências de dígitos binários possibilita a definição e aplicação de operadores genéticos como a recombinação ou a mutação.

Por outro lado, a utilização de uma representação deste tipo faz com que os indivíduos possam ser vistos a dois níveis distintos: ao nível do genótipo e ao nível do fenótipo. O fenótipo de um indivíduo é o seu valor no domínio em que a função objectivo está definida. Ou seja, o fenótipo é a expressão do genótipo, i.e., a estrutura consistindo na codificação das variáveis de decisão através de uma sequência de dígitos binários. Logo, a codificação especifica um mapeamento que transforma uma potencial solução para o problema numa estrutura que pode ser manipulada pelo AG.

3.2.2 Medição do Desempenho

A medição do desempenho consiste na atribuição ou associação de valores aos indivíduos de uma população de acordo com a sua utilidade para a resolução de um problema. A

medição do desempenho é muito importante pois condiciona a eficiência do processo de procura. Em contraste com a selecção determinística utilizada pelas EEs, é usual os AGs basearem-se em mecanismos de selecção probabilísticos. Estes mecanismos de selecção que serão descritos mais adiante, em geral, só podem ser utilizados para valores positivos da medida do desempenho. Por este motivo, é relevante descrever alguns dos possíveis esquemas para medição do desempenho dos indivíduos em AGs.

Todos os mecanismos de medição do desempenho são apresentados para problemas de minimização de uma função objectivo $f(\mathbf{x})$.

Proporcional Simples

Na medição proporcional simples [Gol89], os valores associados aos indivíduos de uma população dependem dos valores da função objectivo $f(\mathbf{x})$:

$$F(\mathbf{x}) = f(\mathbf{x}). \tag{3.2.1}$$

Para além de se ter que garantir que a função objectivo é sempre positiva, este tipo de medição do desempenho tem o inconveniente de, em certos casos, ser causa de convergência prematura, em especial, quando a média das medidas do desempenho dos indivíduos da população se aproximar das medidas do desempenho dos melhores indivíduos. Quando isto acontece, quer os melhores indivíduos quer os indivíduos com medidas do desempenho próximas da média terão aproximadamente a mesma probabilidade de se reproduzirem e gerar descendentes, o que poderá conduzir à convergência prematura do processo de procura. Para evitar este efeito, é possível a implementação de procedimentos de alteração de escala, como a escala linear, a não linear ou a utilização de uma janela de escala.

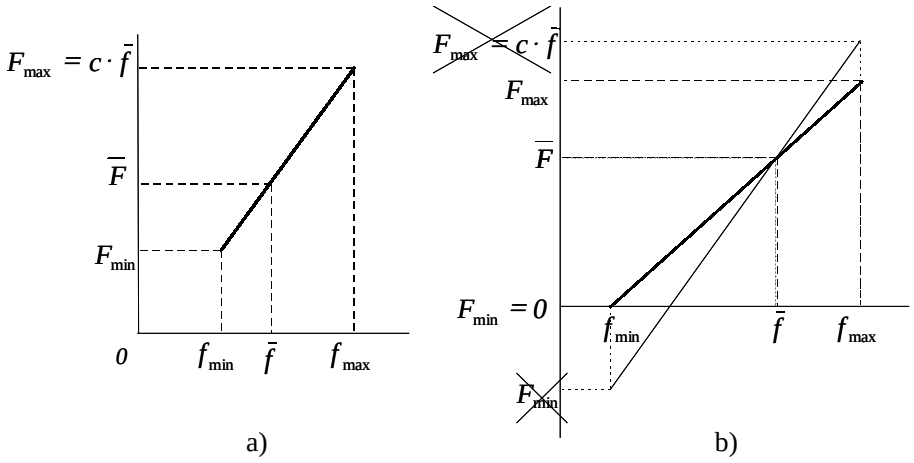

Figura 3.5: Medição do Desempenho com Escala Linear [Gol89]

Escala Linear

A medição do desempenho com escala linear [Gol89] requer a seguinte relação linear entre a função medida do desempenho $F(\mathbf{x})$ e a função objectivo $f(\mathbf{x})$:

$$F(\mathbf{x}) = mf(\mathbf{x}) + b. \tag{3.2.2}$$

Os coeficientes m e b podem ser calculados de diversas formas. No entanto, deve-se garantir que a média dos valores em escala $\overline{F}$ é igual à média dos valores da função objectivo $\overline{f}$. O valor máximo da função medida do desempenho $F_{\max}$ é obtido a partir da seguinte relação:

$$F_{\max} = c\overline{f}, \tag{3.2.3}$$

onde, tipicamente, o coeficiente c é escolhido de tal forma que $c > 1$. Os valores de m e b são calculados da seguinte forma:

$$m = \frac{F_{\max} - \overline{F}}{f_{\max} - \overline{f}} = \frac{c\overline{f} - \overline{f}}{f_{\max} - \overline{f}} = \frac{\overline{f}(c-1)}{f_{\max} - \overline{f}} \text{ e } b = \overline{f} - m\overline{f}. \tag{3.2.4}$$

A Figura 3.5a) ilustra a aplicação de medição do desempenho com escala linear.

Quando o valor de $f_{\min}$ estiver bastante afastado dos valores de $\overline{f}$ e de $f_{\max}$ (situação bastante comum), os valores da função medida do desempenho $F(\mathbf{x})$ podem, indesejavelmente, tornar-se negativos (Figura 3.5b)).

Para evitar que os valores da função medida do desempenho $F(x)$ sejam negativos, deve-se impor que a média dos valores em escala $\overline{F}$ seja igual à média dos valores da função objectivo $\overline{f}$ e $F_{\min} = 0$. Para tal, o valor de m deve ser calculado da seguinte forma:

$$m = \frac{\overline{F} - F_{\min}}{\overline{f} - f_{\min}} = \frac{\overline{F}}{\overline{f} - f_{\min}}. \tag{3.2.5}$$

Janela de Escala

A medição do desempenho utilizando uma janela de escala [Gre84] requer a seguinte relação entre a função medida do desempenho $F(\mathbf{x})$ e a função objectivo $f(\mathbf{x})$:

$$F(\mathbf{x}) = f_{\max} - f(\mathbf{x}), \tag{3.2.6}$$

onde $f_{\max}$ é o valor máximo que a função objectivo $f(\mathbf{x})$ pode tomar num determinado espaço de procura. Esta definição garante que a função medida do desempenho $F(\mathbf{x})$ é sempre positiva quaisquer que sejam as características da função objectivo $f(\mathbf{x})$. A dificuldade reside na determinação do valor de $f_{\max}$ que não é conhecido à partida. Em geral, recorre-se a uma janela de escala cujo tamanho t define o número de gerações imediatamente anteriores à geração actual. A janela de escala permite aproximar o valor de $f_{\max}$, podendo a anterior equação ser reescrita da seguinte forma:

$$F(\mathbf{x}) = \hat{f}_{\max}(t) - f(\mathbf{x}), \tag{3.2.7}$$

onde $\hat{f}_{\max}(t)$ é maior valor obtido para a função objectivo $f(\mathbf{x})$ nas últimas t gerações. O valor do tamanho de janela deve ser inteiro e superior ou igual a 1 (i.e., $t \geq 1$). Muitas vezes, um tamanho de janela infinito (todas as gerações anteriores) é definido para $t = 0$.

Truncagem Sigma

A medição do desempenho por truncagem sigma, proposta por Forrest [For85], utiliza a variância dos valores da função objectivo dos indivíduos de uma população para alterar a escala. Nesta medição, os valores da função medida do desempenho $F(\mathbf{x})$ são obtidos subtraindo-se uma constante aos valores da função objectivo $f(\mathbf{x})$:

$$F(\mathbf{x}) = f(\mathbf{x}) - (\overline{f} - c\sigma). \tag{3.2.8}$$

Nesta equação, a constante c deve ser escolhida adequadamente (tipicamente, $1 \leq c \leq 3$) sendo multiplicada pelo desvio padrão dos valores da função objectivo dos indivíduos da população. O problema dos valores negativos da função medida do desempenho $F(\mathbf{x})$ é contornado fazendo-se $F(\mathbf{x}) = 0$ quando estes ocorrem:

$$F(\mathbf{x}) = \begin{cases} f(\mathbf{x}) - (\overline{f} - c\sigma) & \text{se } f(\mathbf{x}) > \overline{f} - c\sigma \\ 0 & \text{caso contrário} \end{cases} . \tag{3.2.9}$$

Lei de Potência

Este tipo de medição do desempenho, sugerido por Gillies [Gil85], usa uma lei de potência para alterar a escala. Nesta medição, os valores da função medida do desempenho $F(\mathbf{x})$ são obtidos através de uma determinada potência da função objectivo $f(\mathbf{x})$:

$$F(x) = f(x)^k. \tag{3.2.10}$$

No entanto, a determinação do valor para a constante k não é simples, dependendo do problema considerado.

Graduações

Na medição do desempenho baseada em graduações, os indivíduos de uma população são ordenados de acordo com os valores da função objectivo. A medida do desempenho asso-

ciada a cada indivíduo depende apenas da sua ordem e não dos valores da função objectivo [Bak85].

Este tipo de medição do desempenho permite ultrapassar os problemas de escala da medição do desempenho proporcional, comportando-se de uma forma mais robusta [BH91, Whi89]. A estagnação, nos casos em que a pressão de selecção (probabilidade do melhor indivíduo ser seleccionado comparada com a probabilidade média de selecção de todos os indivíduos) é demasiado pequena ou em que ocorre convergência prematura, restringe a procura rapidamente. Com a medição do desempenho baseada em graduações, a gama de indivíduos para reprodução é limitada de tal forma que nenhum indivíduo gera um número excessivo de descendentes, i.e., é possível controlar, de uma forma simples e eficiente, a pressão de selecção.

Na medição baseada em graduações, a medida do desempenho de um indivíduo i, $F(\mathbf{x}_i)$, em que i é inteiro e representa a posição relativa de um indivíduo na população de P indivíduos ($0 \leq i \leq P-1$) após ordenação de acordo com o valor da função objectivo $f(\mathbf{x})$, é calculada pela seguinte relação linear:

$$F(\mathbf{x}_i) = \frac{2(F_{\min}-1)}{P-1} i - F_{\min} + 2. \tag{3.2.11}$$

Nesta equação, i é a posição relativa de um indivíduo na população (os indivíduos com menor e maior valor da função objectivo têm, respectivamente, $i=0$ e $i=P-1$) e $F_{\min}$ corresponde ao menor valor de $F(\mathbf{x}_i)$ ($0 \leq F_{\min} \leq 1$). O valor de $F_{\min}$ define a pressão de selecção.

Tratamento das Restrições

Grande parte das implementações de AGs para optimização com restrições utilizam o método de função de penalização [Gol89], $f_1'(\mathbf{x})$, definida da seguinte forma,

$$f_1'(\mathbf{x}) = f(\mathbf{x}) + R\left(\sum_{j=1}^{m}[\max\{0, g_j(\mathbf{x})\}]^2 + \sum_{i=1}^{p}[h_i(\mathbf{x})]^2\right), \tag{3.2.12}$$

i.e., como a soma de $f(\mathbf{x})$, a função objectivo, e dois termos de penalização para as violações das restrições do tipo igualdade e desigualdade, onde R é um coeficiente de penalização. A utilização de um único coeficiente de penalização na equação (3.2.12) pressupõe que as restrições $g_j(\mathbf{x})$ apresentem a mesma ordem de grandeza. Caso isso não aconteça, é possível normalizar as restrições dividindo, em cada geração, cada restrição $g_j(\mathbf{x})$ pelo maior valor de violação da restrição, G_j, verificado na população, i.e., $g'_j(\mathbf{x}) = g_j(\mathbf{x})/G_j$. O mesmo procedimento pode ser adoptado para as restrições do tipo igualdade.

Esta abordagem tem o grande inconveniente de que, em geral, a escolha do parâmetro de penalização requer muita experimentação. Por este motivo, Deb [Deb00] propôs um método de tratamento de restrições onde é introduzido um termo de penalização que não depende de nenhum coeficiente de penalização. Neste método, a função de penalização, $f'_2(\mathbf{x})$, é definida da seguinte forma,

$$f'_2(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{se } \mathbf{g}(\mathbf{x}) \geq 0 \text{ e } \mathbf{h}(\mathbf{x}) = 0 \\ f_{\max} + \left(\sum_{j=1}^{m} |\max\{0, g_j(\mathbf{x})\}| + \sum_{i=1}^{p} |h_i(\mathbf{x})| \right) & \text{caso contrário} \end{cases}, \tag{3.2.13}$$

onde $f_{\max}$ é valor máximo da função objectivo de todas as soluções admissíveis na população. O valor da função de penalização de qualquer solução não admissível resulta da soma das violações das restrições e o valor da função objectivo mais elevado de todas as soluções admissíveis (f_{max}). Logo, qualquer solução não admissível terá um valor da função de penalização pior do que qualquer uma das soluções admissíveis.

Nesta abordagem, quaisquer duas soluções são comparadas com base,

- nos valores da função objectivo, se admissível;
- nas violações das restrições, se não admissível.

No entanto, deve-se referir que com estas abordagens para o tratamento de restrições, tal como no caso do esquema utilizado pelas EEs, poderão surgir dificuldades na procura

devido à existência de restrições do tipo igualdade. Este facto relaciona-se com a dificuldade em, ao longo da procura, gerar novas soluções admissíveis.

3.2.3 Selecção

A selecção determina quais os indivíduos da população que são escolhidos para progenitores. Aos indivíduos progenitores aplicam-se operadores genéticos gerando novos indivíduos. De notar que, em geral, nos AGs, a selecção é probabilística e é feita de acordo com as medidas do desempenho dos indivíduos de tal forma que os melhores tenham maior probabilidade de serem seleccionados. Assim, o número de descendentes de cada indivíduo seleccionado está relacionado com o mecanismo de selecção considerado. Diversos algoritmos de selecção podem ser implementados. Em seguida, os algoritmos mais frequentemente utilizados são descritos.

Selecção por Roleta

A selecção por roleta é um mecanismo de escolha estocástica. A cada indivíduo associa-se uma fatia da roleta proporcional, em tamanho, à medida do desempenho do indivíduo. Em seguida, gera-se um número aleatório e, o indivíduo cuja fatia inclua esse número é escolhido. O processo é repetido até se seleccionar o número de indivíduos desejado. Este algoritmo pode, também, ser descrito de forma análoga, considerando um segmento de recta cujo comprimento corresponde ao total das medidas do desempenho dos indivíduos da população. Aos indivíduos são associados sub-segmentos de recta contíguos com comprimentos iguais às suas medidas do desempenho. Em seguida, o processo de geração de números aleatórios é repetido o número de vezes desejado e, os indivíduos cujos sub-segmentos de recta incluírem esses números são seleccionados [Bak87].

Selecção por Amostragem Universal Estocástica

Na selecção por amostragem universal estocástica, tal como no algoritmo anterior, é considerado um segmento de recta cujo comprimento corresponde ao total das medidas do desempenho dos indivíduos da população. A cada indivíduo é associado um sub-segmento de recta com comprimento igual à sua medida do desempenho. Em seguida, são determinados pontos do segmento de recta igualmente espaçados, em número igual ao número de indivíduos a seleccionar. Se i for esse número de pontos (número de indivíduos a seleccionar), então o espaçamento entre eles será $\frac{1}{i}$. A posição do primeiro ponto é um número pertencente ao intervalo $[0, \frac{1}{i}]$, determinado de forma aleatória [Bak87].

Selecção por Torneio

Na selecção por torneio, um número t de indivíduos da população é escolhido de forma aleatória e o melhor indivíduo deste grupo, de acordo com as medidas do desempenho, é seleccionado para progenitor. Este processo é repetido tantas vezes quanto o número de indivíduos a escolher. Neste algoritmo é necessário definir o valor do parâmetro t [GD91]. O valor deste parâmetro está relacionado com a pressão de selecção. Quanto maior é o valor do parâmetro t, maior é a pressão de selecção.

Selecção por Truncagem

Na selecção por truncagem, os indivíduos são ordenados de acordo com a sua medida do desempenho. Após a ordenação, só os melhores indivíduos são seleccionados para serem progenitores. Neste algoritmo é necessário definir a proporção de indivíduos da população a serem seleccionados para progenitores [CK70].

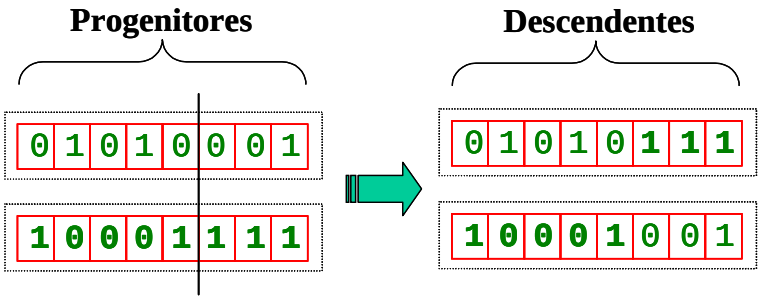

Figura 3.6: Cruzamento Simples

3.2.4 Recombinação

A recombinação ou cruzamento produz novos indivíduos (descendentes) combinando a informação dos seus progenitores. Considerando que os indivíduos são representados por sequências de dígitos binários, diferentes algoritmos de recombinação podem ser aplicados. Os mais frequentemente utilizados são descritos em seguida.

Cruzamento Simples

No cruzamento simples determina-se, de forma aleatória (uniformemente), um ponto de cruzamento. A troca de informação a partir desse ponto entre os indivíduos progenitores produz dois descendentes. A Figura 3.6 exemplifica a aplicação deste operador genético considerando o ponto de cruzamento na posição 5.

Cruzamento Múltiplo

No cruzamento múltiplo determinam-se, de forma aleatória (uniformemente), i pontos de cruzamento. Depois de ordenados por ordem crescente os pontos de cruzamento, entre cada dois pontos de cruzamento consecutivos, troca-se informação entre os indivíduos progenitores produzindo dois descendentes. A informação até ao primeiro ponto de cruzamento não é trocada entre os dois indivíduos progenitores [Boo87]. Na Figura 3.7 é exemplificada

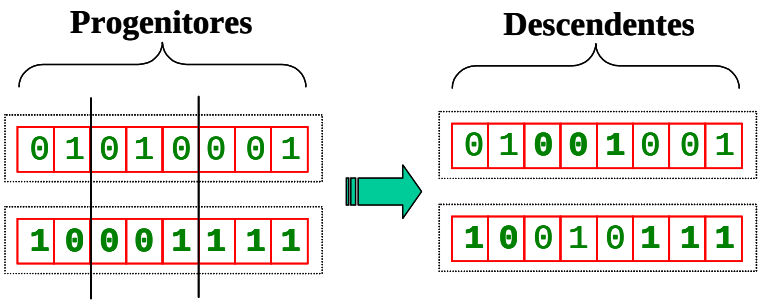

Figura 3.7: Cruzamento Múltiplo com 2 Pontos de Cruzamento

a aplicação deste operador genético considerando dois pontos de cruzamento nas posições 2 e 5 (cruzamento duplo).

O cruzamento múltiplo possibilita a diminuição dos fenómenos de ruptura do cruzamento simples. A ruptura é um fenómeno que consiste em, durante o processo de recombinação genética, certa informação dos indivíduos progenitores não ser passada aos indivíduos descendentes. Na prática, pode-se dizer que a redução deste efeito de ruptura conduz a um aumento da eficiência do mecanismo de cruzamento. De Jong [DJ75] referiu e, mais tarde, os trabalhos de Booker [Boo82] mostraram que este fenómeno pode ser reduzido pela alteração do mecanismo de cruzamento. A alteração referida consiste simplesmente em considerar mais de um ponto de cruzamento seleccionado aleatoriamente.

Cruzamento Uniforme

No cruzamento uniforme, todos os pontos são potencialmente pontos de cruzamento. Uma máscara é gerada de forma aleatória, e a paridade dos dígitos binários da máscara indica qual progenitor e que dígitos binários fornecerá aos descendentes [Sys89]. A Figura 3.8 ilustra a aplicação do cruzamento uniforme em que a máscara binária considerada é 0 1 1 0 0 0 0 1.

Tal como o cruzamento múltiplo, o cruzamento uniforme permite diminuir o efeito de

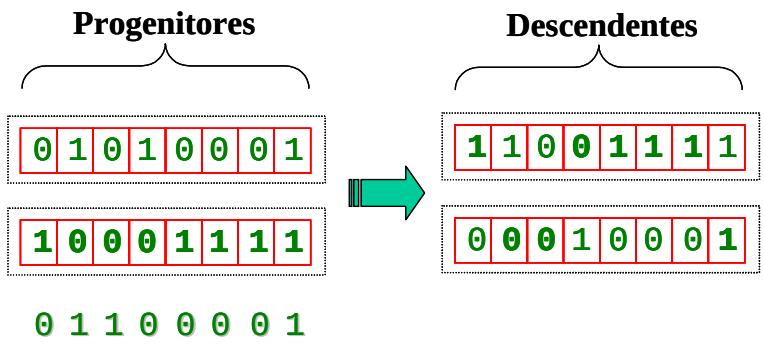

Figura 3.8: Cruzamento Uniforme

ruptura. Para além disso, Spears e De Jong [SDJ91] demonstraram que o cruzamento uniforme pode ser parametrizado atribuindo uma determinada probabilidade para a troca de dígitos binários. Este parâmetro permite controlar a quantidade de ruptura na recombinação.

3.2.5 Mutação

A mutação é um operador unário que consiste em perturbar ligeiramente (com uma probabilidade pequena) os indivíduos descendentes gerados pela recombinação. Considerando que os indivíduos são representados por sequências de dígitos binários, cada dígito binário é, de forma aleatória, trocado de 0 para 1 ou vice-versa, de acordo com uma determinada probabilidade. A mutação permite garantir que um determinado caracter genético não tende a desaparecer na população. A Figura 3.9 exemplifica a aplicação de mutação uniforme em que o dígito binário na posição 2 é mutado.

3.2.6 Re-inserção

Os novos indivíduos gerados devem ser inseridos na população. Para isso, é necessário, entre outras coisas, ter em consideração o número indivíduos descendentes e qual a política

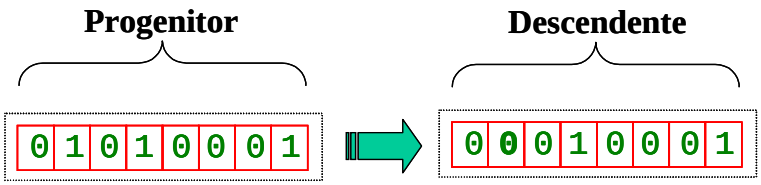

Figura 3.9: Mutação Uniforme

de escolha dos indivíduos da população que irão ser substituídos. Alguns dos possíveis esquemas de substituição são descritos em seguida.

Pura

Na re-inserção pura, o número de indivíduos descendentes gerados é igual ao tamanho da população. No final de uma geração, a nova população será constituída exclusivamente pelos indivíduos descendentes. Este é o esquema de re-inserção utilizado pelo AG na sua forma mais simples.

Uniforme

Na re-inserção uniforme, o número de indivíduos descendentes gerados é inferior ao tamanho da população. Os indivíduos descendentes vão substituir indivíduos da população escolhidos de forma aleatória.

Elitista

Este esquema proposto inicialmente por De Jong [DJ75] garante que os melhores indivíduos encontrados ao longo da procura se encontram presentes na população na última geração. Na re-inserção elitista, define-se uma elite constituída pelos melhores indivíduos encontrados até ao momento. Logo, o número de indivíduos descendentes gerados é inferior ao tamanho da população. Os indivíduos descendentes vão substituir os indivíduos da po-

pulação que não fazem parte da elite. Em geral, os melhores θ indivíduos da população são escolhidos para formar a elite. A escolha do parâmetro θ está directamente relacionada com a pressão de selecção. Quando valores elevados de θ são escolhidos, existe uma maior influência da elite na procura e a diversidade da população tende a diminuir. Tipicamente, valores de $\theta = 1$ ou $\theta = 0.1P$ (10% da população) são utilizados.

3.2.7 Controlo da Diversidade

Com o objectivo de lidar com problemas de optimização multi-modais onde, muitas vezes, se pretende encontrar diferentes soluções, torna-se necessário implementar mecanismos de controlo da diversidade na população. Um dos mecanismos de controlo da diversidade mais utilizado é o da partilha da medida do desempenho (*sharing*) sugerido por Goldberg e Richardson [GR87] e, mais tarde, estudado por Deb e Goldberg [DG89]. Este mecanismo promove a formação e manutenção de sub-populações de indivíduos, os nichos, com o objectivo de se encontrar as diversas soluções pretendidas, e baseia-se na ideia de que os indivíduos de um determinado nicho devem partilhar os recursos, i.e., a medida do desempenho. Assim, a medida do desempenho de um dado indivíduo é degradada na proporção do número de indivíduos que estejam localizados na sua vizinhança. A noção de vizinhança de um indivíduo implica a especificação de um parâmetro, o σ_{share}, que define o raio dos nichos. Para a determinação dos indivíduos contidos numa dada vizinhança é necessária a utilização de uma medida da distância, $d(\mathbf{x}_i, \mathbf{x}_j)$, de dois indivíduos $\mathbf{x}_i$ e $\mathbf{x}_j$ (em geral, a distância euclidiana é considerada). Matematicamente, a função medida do desempenho de *sharing*, $F_{sharing}(\mathbf{x}_i)$ para um indivíduo $\mathbf{x}_i$ com $0 \leq i \leq P-1$ é dada por:

$$F_{sharing}(\mathbf{x}_i) = \frac{F(\mathbf{x}_i)}{\sum_{j=0}^{P-1} Sh(d(\mathbf{x}_i, \mathbf{x}_j))}, \tag{3.2.14}$$

onde $Sh(d(\mathbf{x}_i, \mathbf{x}_j))$ é uma função de *sharing*. Uma função de *sharing* muito utilizada é

$$Sh(d(\mathbf{x}_i, \mathbf{x}_j)) = \begin{cases} 1 - \left(\frac{d(\mathbf{x}_i, \mathbf{x}_j)}{\sigma_{share}}\right)^2 & \text{se } d(\mathbf{x}_i, \mathbf{x}_j) \leqslant \sigma_{share} \\ 0 & \text{caso contrário} \end{cases} . \tag{3.2.15}$$

Em geral, a distância entre dois indivíduos, $\mathbf{x}_i$ e $\mathbf{x}_j$, é medida ao nível das variáveis de decisão, i.e., $d(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$.

3.2.8 Algoritmos Genéticos com Representação Real

Apesar de tradicionalmente os AGs utilizarem representações binárias das variáveis de decisão, mais recentemente, têm surgido diversas implementações que utilizam representações reais. Uma das primeiras implementações foi proposta por Wright [Wri91], que desenvolveu operadores genéticos para lidar com a representação real das variáveis (e.g., o cruzamento linear). Implementações mais recentes foram propostas por Deb e Agrawal [DA95] e Deb e Kumar [DK95], incluindo novos operadores genéticos (e.g., o cruzamento SBX -*simulated binary crossover*).

3.2.9 Alguns Fundamentos Teóricos

Nesta secção são abordados os fundamentos teóricos dos AGs com representação binária.

Teorema Fundamental dos Algoritmos Genéticos

Dada a natureza probabilística do algoritmo, não pode ser provado que, na última geração, a população de indivíduos contenha uma ou a solução óptima, mas pode ser mostrado matematicamente que, em certas condições, o AG optimiza a geração de novos indivíduos (como será mostrado mais adiante). Este fundamento matemático dos AGs é mostrado pelo Teorema Fundamental dos Algoritmos Genéticos de Holland [Hol75, Gol89].

Começa-se pela apresentação do conceito de esquema que é essencial à compreensão do Teorema Fundamental dos Algoritmos Genéticos. Para isso, considere-se uma população de

P indivíduos e em que cada indivíduo é uma sequência de dígitos de tamanho l definida no alfabeto $\{0, 1\}$ (alfabeto binário). Define-se *esquema* como sendo uma sequência de dígitos no alfabeto $\{0, 1, *\}^l$, onde o caracter $*$ é um meta-caracter que corresponde a qualquer caracter do alfabeto considerado.

Um esquema E representa o conjunto de todos os indivíduos obtidos a partir de E substituindo por 0 ou 1 (tratando-se do alfabeto binário), o caracter $*$ de todas as formas possíveis. O comprimento $C(E)$ de um esquema é definido como a distância entre os seus dois caracteres mais distantes diferentes do caracter $*$. A ordem $O(E)$ de um esquema é definida como o seu número total de caracteres menos o seu número de caracteres $*$.

Por exemplo, seja $E = (0 * * * 10*)$ um esquema de tamanho $l = 7$ duma população de indivíduos. O esquema E representa o conjunto de $2^4 = 16$ indivíduos distintos, sendo o comprimento e a ordem de E, respectivamente, $C(E) = 6 - 1 = 5$ e $O(E) = 3$.

Para um mesmo tamanho fixo l de uma sequência de dígitos, o número de esquemas distintos depende da cardinalidade do alfabeto considerado. O número de esquemas distintos, para o alfabeto binário será 3^l, e, mais genericamente, $(c + 1)^l$ para o caso de um qualquer alfabeto de cardinalidade c. Logo, o alfabeto binário é o alfabeto que possibilita o maior número de esquemas por *bit* de informação na codificação de sequências de dígitos.

Para ilustrar este princípio, considere-se o problema da codificação de sequências de dígitos de tamanho l, usando o alfabeto binário ($\{0, 1\}$) ou, alternativamente, o alfabeto de cardinalidade 4 (e.g.: $\{A, B, C, D\}$). Se se pretender representar 4 valores distintos, usando cada um destes alfabetos, é fácil verificar que no caso do alfabeto binário, é necessário usar sequências de dígitos de tamanho $l = 2$, bastando no caso do alfabeto de cardinalidade 4, usar sequências de dígitos de tamanho $l = 1$. No primeiro caso, o número de esquemas é $3^2 = 9$, e no segundo é $5^1 = 5$. Com este exemplo limite, é fácil verificar que o número de esquemas no alfabeto binário é superior ao número de esquemas em alfabetos de cardinalidade superior. Note-se, também, que usando os dois códigos pode-se representar

o mesmo número de valores distintos; no entanto, alfabetos com diferentes cardinalidades requerem sequências de dígitos de diferentes tamanhos. Por outro lado, a representação binária pode obrigar ao sacrifício de precisão à medida que o domínio de valores a codificar aumenta e se impõe um tamanho de sequências de dígitos cujo tratamento seja exequível.

Teoricamente, os AGs são usualmente analisados como algoritmos que processam esquemas, i.e., descrições de hiperplanos em um espaço n-dimensional de dígitos binários [Gol89]. O Teorema Fundamental dos Algoritmos Genéticos mostra que, nos AGs, os esquemas curtos e de baixa ordem, os chamados blocos construtivos, tendem a aumentar exponencialmente em número [Hol75].

Efeito da Selecção Considere-se uma população de P indivíduos num instante de tempo t e seja $A(E,t)$ o número de indivíduos do esquema E na população. A aplicação do operador selecção a uma população de indivíduos faz com que o número de ocorrências de E se modifique de acordo com a equação:

$$A(E,t+1) = A(E,t)P\frac{F(E,t)}{\sum_{j=1}^{P} F_j(t)} = A(E,t)\frac{F(E,t)}{\overline{F}(t)}, \tag{3.2.16}$$

onde $F(E,t)$ representa uma estimativa da medida do desempenho do esquema E (e.g., esta estimativa pode ser obtida calculando-se a média das medidas de desempenho de todos os indivíduos da população que pertencem ao esquema E) e é a média das medidas de desempenho dos indivíduos da população.

A equação (3.2.16) significa que o número de indivíduos na população pertencentes ao esquema E aumenta com a razão da medida do desempenho do esquema E e $\overline{F}(t)$, a média das medidas de desempenho dos indivíduos da população; i.e., o número de indivíduos para um dado esquema com medida do desempenho superior à média tende a aumentar. Por outro lado, esse número tende a diminuir, se a medida do desempenho do esquema for inferior à média, e tende a manter-se, se a medida do desempenho for igual à média.

Para além disso, se for assumido que a medida do desempenho do esquema E se mantém ao longo das gerações, acima da média das medidas de desempenho dos indivíduos da população, de uma quantidade R constante[1] (i.e., $F(E,t) = R\overline{F}(t)$), a equação (3.2.16) pode ser vista como a equação de uma progressão geométrica de razão R:

$$A(E,t+1) = A(E,0)R^{t+1} \text{ em que } R = \frac{F(E,t)}{\overline{F}(t)}. \tag{3.2.17}$$

Na equação (3.2.17), o valor de R será superior ou inferior a 1, consoante a medida do desempenho do esquema E esteja acima ou abaixo da média das medidas de desempenho dos indivíduos da população (será nulo se a medida do desempenho do esquema E for igual à média das medidas de desempenho dos indivíduos da população).

Neste momento, pode-se não só afirmar que esquemas cuja, medida do desempenho esteja acima da média das medidas de desempenho dos indivíduos da população, aumentarão em número na geração seguinte, mas também que aumentarão exponencialmente, em número, nas gerações seguintes.

Efeito da Recombinação Considerando o cruzamento com um ponto escolhido de acordo com uma probabilidade uniformemente distribuída, em que este, destrói o esquema E de comprimento $C(E)$ com a probabilidade:

$$\Pr_{dest}(E) = \frac{C(E)}{l-1}. \tag{3.2.18}$$

Consequentemente, o esquema E de comprimento $C(E)$ sobrevive com a probabilidade:

$$\Pr_{sobrev}(E) = 1 - \frac{C(E)}{l-1}. \tag{3.2.19}$$

[1]Esta suposição, embora razoável para um determinado conjunto de esquemas, não é verdadeira, uma vez que a medida do desempenho de um esquema E e a média das medidas de desempenho dos indivíduos da população variam ao longo das gerações (i.e., $R(t) = \frac{F(E,t)}{\overline{F}(t)}$).

Considerando que a recombinação é aplicada de acordo com uma probabilidade de cruzamento $\Pr_{cruz}$, o esquema E sobrevive à recombinação com probabilidade:

$$\Pr_{sobrev}(E) = 1 - \Pr_{cruz} \frac{C(E)}{l-1}. \tag{3.2.20}$$

De notar que, durante a recombinação, se o ponto de cruzamento se situar entre posições fixas do esquema[2], existe a possibilidade dele sobreviver (embora com uma probabilidade pequena). Pelo que, a equação (3.2.20) deve ser modificada:

$$\Pr_{sobrev}(E) \geq 1 - \Pr_{cruz} \frac{C(E)}{l-1}. \tag{3.2.21}$$

Finalmente, combinando os efeitos da selecção e da recombinação no esquema E obtém-se:

$$A(E, t+1) \geq A(E, t) \frac{F(E,t)}{\overline{F}(t)} (1 - \Pr_{cruz} \frac{C(E)}{l-1}) \tag{3.2.22}$$

que denota o número de indivíduos do esquema E na população após a recombinação.

Efeito da Mutação Seja $\Pr_{mut}$ a probabilidade de mutação de um caracter específico (diferente do caracter $*$) de um esquema E com ordem $O(E)$. O esquema E sobreviverá à aplicação da mutação com probabilidade:

$$\Pr_{sobrev}(E) = (1 - \Pr_{mut})^{O(E)}. \tag{3.2.23}$$

No caso de $\Pr_{mut}$ ter um valor bastante pequeno, esta probabilidade pode ser aproximada pela expressão:

$$\Pr_{sobrev}(E) \cong 1 - O(E) \Pr_{mut}. \tag{3.2.24}$$

Os efeitos da selecção, da recombinação e da mutação podem agora ser expressos da seguinte forma:

$$A(E, t+1) \geq A(E, t) \frac{F(E,t)}{\overline{F}(t)} (1 - \Pr_{cruz} \frac{C(E)}{l-1} - O(E) \Pr_{mut}). \tag{3.2.25}$$

[2]Posições onde os caracteres do esquema são diferentes de $*$.

A equação (3.2.25) descreve o Teorema Fundamental dos Algoritmos Genéticos e, essencialmente, expressa que esquemas curtos (pequeno comprimento), de baixa ordem, os chamados blocos construtivos, com medida do desempenho maior ou igual à média das medidas de desempenho dos indivíduos da população, aumentarão exponencialmente em número.

No entanto, mais recentemente, autores como Vose [Vos99] e Mühlenbein e Mahnig [MM02] questionaram a validade do Teorema Fundamental dos Algoritmos Genéticos. Para alguns problemas de optimização, o teorema anterior parece não ser válido. Mühlenbein e Mahnig [MM02] apresentam um exemplo de um problema onde a proliferação de esquemas ao longo das gerações não ocorre de acordo com o Teorema Fundamental dos Algoritmos Genéticos. Logo, apesar de para um grande número de problemas de optimização, ao longo da procura, os esquemas variarem em número de acordo com o teorema, parece haver algumas classes de problemas em que isso não se verifica.

Representação

A codificação de sequências de dígitos binários pode ser realizada recorrendo a diferentes alfabetos e a diferentes mecanismos de codificação. Segundo Goldberg [Gol89], devem ser tidos em conta dois princípios básicos na escolha do alfabeto e do mecanismo de codificação a usar num determinado problema: o Princípio dos Blocos Construtivos Significativos e o Princípio do Alfabeto Mínimo.

De acordo com o primeiro princípio, deve-se escolher um código tal que esquemas curtos (pequeno comprimento) e de baixa ordem, os blocos construtivos, sejam relevantes para o problema em causa e não relacionados com outros esquemas através de outras posições fixas (de dígitos binários). Logo, quando se escolhe um código deve-se ter em conta a distância entre dígitos binários relacionados (muitas vezes, a reordenação dos dígitos binários poderá possibilitar a obtenção de um "melhor" código).

O segundo princípio postula que se deve escolher de entre os alfabetos possíveis, aquele que possibilite uma descrição mais simples do problema considerado.

Em todas as aplicações de AGs é necessário decidir qual a forma de representação dos indivíduos. As representações binárias são as mais frequentes e, em termos precisos, o fundamento matemático dos AGs, o Teorema Fundamental dos Algoritmos Genéticos, foi apenas mostrado para este tipo de representação.

3.3 Comparação Empírica de Estratégias Evolutivas e Algoritmos Genéticos

Nesta secção são considerados os dois Algoritmos Evolucionários (AEs) mais frequentemente aplicados a problemas de optimização: os Algoritmos Genéticos (AGs) e as Estratégias Evolutivas (EEs). Estes AEs têm características diferentes, nomeadamente na forma como tratam as restrições e na representação das soluções. Logo, é importante comparar o seu desempenho quando aplicados a diferentes tipos de problemas de optimização. A identificação das vantagens e desvantagens de cada abordagem para diferentes tipos de problemas de optimização tem um forte interesse prático para conhecer o desempenho esperado destas abordagens quando aplicadas a um problema de optimização particular.

Com o objectivo de comparar estas duas abordagens, diversos problemas de Programação Não Linear (PNL) com diferentes características foram considerados [CO98].

3.3.1 Casos de Estudo

Todos os problemas considerados possuem variáveis de decisão contínuas. A Tabela 3.2 apresenta a lista dos problemas de PNL [Sch95, HS81, FP87] considerados:

- Não lineares, multi-modais sem restrições (A1, ..., A7),

- Não lineares, uni-modais sem restrições (B1, ..., B9), e
- Não lineares, uni-modais com restrições (C1, ..., C4).

Nesta tabela, n é o número de variáveis de decisão do problema e m é o número de restrições do tipo desigualdade. As funções objectivo a minimizar e as restrições dos problemas são também apresentadas. Para todos os problemas considerados, o óptimo global é conhecido à priori. Portanto, o sucesso do AG ou das EEs pode ser avaliado medindo, por exemplo, a distância do resultado obtido à solução óptima (e.g., um vector solução $\mathbf{x}$ pode ser considerado na vizinhança do óptimo global $\mathbf{x}^*$, quando cada variável x_i de $\mathbf{x}$ está a menos de ϵ_i do óptimo global x_i^*, i.e., $|x_i - x_i^*| \leq \epsilon_i$ para todo o $i = 1, \ldots, n$).

3.3.2 Resultados Computacionais

As Tabelas 3.3 e 3.4 resumem os valores dos parâmetros utilizados nas execuções do AG e das EEs. Todos os parâmetros foram mantidos constantes para todos os problemas com objectivo de focar a análise na dependência do desempenho dos algoritmos no tipo de problema a ser resolvido. Não houve a preocupação de encontrar os melhores valores dos parâmetros dos algoritmos para cada um dos problemas. Pelo contrário, foram escolhidos valores típicos para os parâmetros para estudar o comportamento de cada algoritmo quando aplicado a diferentes problemas.

Para efectuar as comparações foram consideradas quatro variantes de AEs: um AG, uma EE-$(1 + 1)$, uma EE-$(10, 100)$ e uma EE-$(10 + 100)$. As implementações dos AEs foram feitas de acordo com os algoritmos apresentados nas Secções 3.1 e 3.2.

No que diz respeito ao AG, a população inicial de cromossomas foi gerada de forma completamente aleatória. Para representar as variáveis de decisão foram utilizadas sequências de dígitos binários (32 bits por variável de decisão). O mecanismo de selecção considerado foi a selecção universal estocástica [Bak87]. Como operadores genéticos, considerou-se a

#	Problema	n	m	Função Objectivo e Restrições
A1	Shubert	2	-	$f(\mathbf{x}) = \sum_{i=1}^{5} (i\cos((i+1)x_1 + i)) \sum_{i=1}^{5} (i\cos((i+1)x_2 + i))$
A2	Easom	2	-	$f(\mathbf{x}) = -\cos(x_1)\cos(x_2)\exp(-\sum_{i=1}^{2}(x_i - \pi)^2)$
A3	Bohachevsky#1	2	-	$f(\mathbf{x}) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$
A4	Bohachevsky#2	2	-	$f(\mathbf{x}) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3$
A5	Bohachevsky#3	2	-	$f(\mathbf{x}) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) + \cos(4\pi x_2) + 0.3$
A6	Colville	4	-	$f(\mathbf{x}) = 100(x_2 + x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + +10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$
A7	Beale	2	-	$f(\mathbf{x}) = (1.5 - x_1(1 - x_2))^2 + (2.25 - x_1(1 - x_2^2))^2 + +(2.625 - x_1(1 - x_2^3))^2$
B1	Schwefel#1	5	-	$f(\mathbf{x}) = \sum_{i=1}^{5} ((x_1 - x_i^2) + (x_i - 1))$
B2	Booth	2	-	$f(\mathbf{x}) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
B3	Powell	4	-	$f(\mathbf{x}) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 + 2x_3)^4 + 10(x_1 - x_4)^2$
B4	Leon	2	-	$f(\mathbf{x}) = 100(x_2 - x_1^3)^2 + (x_1 - 1)^2$
B5	Schwefel#2	5	-	$f(\mathbf{x}) = \sum_{i=1}^{5} \lvert x_i \rvert$
B6	Schwefel#3	5	-	$f(\mathbf{x}) = \min_{1 \le i \le 5} (\lvert x_i \rvert)$
B7	Schwefel#4	5	-	$f(\mathbf{x}) = \sum_{i=1}^{5} \lvert x_i \rvert + \prod_{i=1}^{5} \lvert x_i \rvert$
B8	Schwefel#5	5	-	$f(\mathbf{x}) = \sum_{i=1}^{5} x_i^10$
B9	Rosenbrock#1	2	-	$f(\mathbf{x}) = 100(x_2 - x_1^2) + (x_1 - 1)^2$
C1	Rosenbrock#2	3	4	$f(\mathbf{x}) = -x_1x_2x_3$ sujeito a $\mathbf{g}(\mathbf{x}) \ge 0$ com $g_j(\mathbf{x}) = \begin{cases} x_j & \text{se } 1 \le j \le 3 \\ -x_1 - 2x_2 - 2x_3 + 72 & \text{se } j = 4 \end{cases}$
C2	Corridor	3	7	$f(\mathbf{x}) = -\sum_{i=1}^{3} x_i$ sujeito a $\mathbf{g}(\mathbf{x}) \ge 0$ com $g_j(\mathbf{x}) = \begin{cases} -x_j + 100 & \text{se } 1 \le j \le 3 \\ x_{j-2} - \frac{1}{j-2} + \sum_{i=1}^{j-3} x_i + \sqrt{\frac{j-1}{j-2}} & \text{se } 4 \le j \le 5 \\ -x_{j-4} + \frac{1}{j-4} + \sum_{i=1}^{j-5} x_i + \sqrt{\frac{j-3}{j-4}} & \text{se } 6 \le j \le 7 \end{cases}$
C3	Schwefel#6	2	1	$f(\mathbf{x}) = \sum_{i=1}^{2} x_i^2$ sujeito a $g_1(\mathbf{x}) = x_1 + 2x_2 - 2 \ge 0$
C4	Wolfe	3	3	$f(\mathbf{x}) = \frac{4}{3}((\sum_{i=1}^{2} x_i^2) - x_1x_2)^{\frac{3}{4}} + x_2)$ sujeito a $\mathbf{g}(\mathbf{x}) \ge 0$ com $g_j(\mathbf{x}) = x_j$ para $1 \le j \le 3$

Tabela 3.2: Casos de Estudo - Problemas de Programação Não Linear

Algoritmo Genético	
Número de experiências	10
Tamanho da população	100
Tamanho do cromossoma	$32n$
Probabilidade de recombinação	0.7
Probabilidade de mutação	0.001
Coeficiente de penalização	100
Mecanismo de codificação	Código *Gray*

Tabela 3.3: Parâmetros do Algoritmo Genético

recombinação com dois pontos (cruzamento duplo) aplicado com uma probabilidade de 0.7 e a mutação uniforme aplicada com uma probabilidade de 0.001. No tratamento das restrições do tipo desigualdade, um esquema de penalização foi implementado, fixando-se o coeficiente de penalização em 100. Estes parâmetros foram mantidos para todos os problemas. O critério de paragem adoptado foi terminar a procura quando um número máximo de 5000 gerações fosse atingido ou todos os cromossomas da população convergissem para um ponto, i.e., quando todos os cromossomas da população fossem iguais.

Nas EEs, dependendo do problema, uma aproximação inicial ao óptimo global foi considerada. Os valores iniciais para os desvios padrão σ_i foram determinados pela equação (4.2.2), assumindo-se $\Delta x = 1$. No caso da EE-$(1+1)$ utilizou-se a "Regra de 1/5 Sucessos" (com $c_{dec} = 0.85$ e $c_{inc} = 1/0.85$) para actualização dos desvios padrão σ_i. Para as EEs multi-membros foi utilizado um esquema de adaptação isotrópica com $\Delta\sigma = n^{-1/2}$. Para lidar com as restrições, um esquema de eliminação foi considerado, i.e., as solução não admissíveis foram eliminadas. O critério de paragem adoptado foi terminar a procura quando o número máximo de 2000 gerações fosse atingido ou as condições de paragem sugeridas por Schwefel [Sch95] para a EE-$(1+1)$ (com $\varepsilon_3 = 10^{-5}$, $\varepsilon_4 = 10^{-5}$ e $\Delta k = 20n$) e para as

Estratégia Evolutiva	$(1+1)$	$(10,100)$	$(10+100)$
Número de experiências	10		
Aproximação inicial	Variável		
Valor inicial de σ	Variável		
Factor de actualização de σ	0.85	$n^{-1/2}$	$n^{-1/2}$
Limite inferior de σ (absoluto)	10^{-5}		
Limite inferior de σ (relativo)	10^{-5}		

Tabela 3.4: Parâmetros das Estratégias Evolutivas

EEs multi-membros fossem verificadas (com $\varepsilon_3 = 10^{-5}$ e $\varepsilon_4 = 10^{-5}$).

Note-se que, para o AG e para as EEs multi-membros, os tamanhos das populações foram semelhantes (100 indivíduos). O número de experiências foi de 10 para todos os algoritmos, i.e., cada algoritmo foi aplicado 10 vezes a cada problema.

A Tabela 3.5 apresenta os resultados das experiências executadas para os problemas considerados. Nesta tabela, $\overline{\#G}$ é o número médio de gerações nas 10 experiências, $C\%$ é a percentagem de vezes em que foi encontrado o óptimo global e $\mathbf{x}^{(0)}$ é a aproximação inicial requerida pelas EEs para gerar a população inicial. Por geração, para o AG e as EEs multi-membros, a função objectivo foi avaliada 100 vezes, pelo que, o número de cálculos da função objectivo é proporcional ao número de gerações.

3.3.3 Discussão dos Resultados

Não foram testados os efeitos dos diversos parâmetros dos algoritmos, uma vez que apenas se pretendia investigar e identificar as vantagens e desvantagens para um conjunto restrito de problemas de optimização. De salientar que os resultados obtidos empiricamente para este conjunto de problemas apenas permitem, de forma limitada, fazer considerações muito genéricas acerca dos AEs considerados. As comparações podem ser feitas em termos de

	AG			EE-(1+1)		EE-(10,100)		EE-(10+100)	
#	$\overline{\#G}$	$C\%$	$\mathbf{x}^{(0)}$	$\overline{\#G}$	$C\%$	$\overline{\#G}$	$C\%$	$\overline{\#G}$	$C\%$
A1	308	**100**	(0,0)	223	**80**	102	**100**	46	**100**
A2	337	**70**	(0,0)	187	**90**	47	**90**	30	**100**
A3	280	**80**	(1,1)	295	**70**	104	**100**	47	**100**
A4	722	**60**	(1,1)	235	**40**	70	**100**	45	**100**
A5	377	**30**	(1,1)	187	**10**	105	**0**	56	**0**
A6	5000	**20**	(0,0,0,0)	2000	**20**	1003	**80**	328	**100**
A7	2171	**100**	(0,0)	1251	**70**	113	**100**	34	**100**
B1	4186	**100**	(0,0,0,0,0)	709	**100**	103	**100**	132	**100**
B2	1001	**100**	(0,0)	343	**100**	72	**100**	34	**100**
B3	5000	**100**	(3,-1,0,1)	2000	**100**	160	**100**	101	**100**
B4	2925	**100**	(0,0)	2000	**100**	204	**100**	104	**100**
B5	5000	**100**	(10,10,10,10,10)	909	**100**	2000	**100**	384	**100**
B6	5000	**100**	(5,5,5,5,5)	1159	**80**	798	**90**	211	**100**
B7	5000	**100**	(10,10,10,10,10)	1259	**40**	2000	**100**	428	**100**
B8	5000	**100**	(10,10,10,10,10)	949	**100**	36	**100**	24	**100**
B9	2273	**80**	(0,0)	2000	**100**	183	**100**	70	**100**
C1	5000	**100**	(0,0,0)	1396	**100**	89	**100**	55	**100**
C2	4665	**40**	(0,0,0)	689	**100**	107	**100**	106	**100**
C3	1623	**90**	(0,0)	243	**100**	106	**100**	59	**100**
C4	806	**100**	(0,1,0)	1453	**100**	110	**100**	46	**100**

Tabela 3.5: Resultados Computacionais

dois aspectos principais: a precisão e a eficiência dos algoritmos. O primeiro aspecto está relacionado com a capacidade dos algoritmos encontrarem uma boa aproximação ao óptimo global de um problema. O segundo refere-se ao esforço computacional requerido para o conseguir (aqui medido em termos do número de cálculos da função objectivo).

Nos problemas multi-modais sem restrições, os resultados do AG e da EE-$(1+1)$ não são muito bons. As EEs multi-membros tiveram um desempenho muito bom em todos os problemas com excepção de um. No entanto, de notar que o mau desempenho do AG neste tipo de problemas pode ser explicado pela escolha do valor da probabilidade de mutação que poderá não garantir a exploração eficiente do espaço de procura de natureza multi-modal.

Nos problemas uni-modais sem restrições, tanto o AG como a EE-$(1+1)$ apresentaram um bom desempenho, convergindo quase sempre para o óptimo global. No entanto, os melhores resultados foram obtidos pelas EEs multi-membros que tiverem um bom desempenho em todos os problemas considerados.

Nos problemas uni-modais com restrições, o AG teve um desempenho razoável (de notar que o mesmo valor do coeficiente de penalização foi usado para todos os problemas); contudo, os melhores resultados foram obtidos pelas EEs que tiverem um bom desempenho em todos os problemas considerados. Estas diferenças podem resultar das diferentes estratégias, utilizadas pelos algoritmos, para lidar com as restrições. De notar que, face ao esquema de tratamento das restrições utilizado pelas EEs, estas teriam dificuldades em lidar com problemas altamente restringidos, o que não era o caso dos problemas considerados.

Em termos genéricos, o número de gerações e de cálculos da função objectivo requeridos pelas EEs foi inferior ao requerido pelo AG. De referir que Bäck [Bäc96] apresenta resultados similares (em termos da eficiência) quando compara três AEs (uma EE, um algoritmo baseado em PE e um AG) aplicados a um conjunto de cinco problemas teste. Em termos

globais, os resultados obtidos permitem fazer a seguinte ordenação dos AEs (por ordem decrescente): EE-(10+100), EE-(10,100), AG e EE-(1+1). Obviamente, existirão outros conjuntos de problemas em que esta ordem poderá ser diferente.

Capítulo 4

Aplicações

Neste capítulo são apresentadas diversas aplicações de Algoritmos Evolucionários (AEs) a problemas de engenharia. Na Secção 4.1 é descrita a aplicação de AEs a um conjunto de problemas de Programação Inteira Mista Não Linear (PIMNL) da área da engenharia química. Em seguida, na Secção 4.2 é exposta a aplicação de um Algoritmo Genético (AG) ao projecto de placas laminadas com materiais isotrópicos. Finalmente, a aplicação de um AG ao projecto de placas laminadas com materiais compósitos é apresentada na Secção 4.3.

4.1 Optimização de Problemas Inteiros Mistos Não Lineares

Muitos problemas do mundo real podem, em geral, ser formulados como problemas de Programação Inteira Mista Não Linear (PIMNL). A formulação matemática de um problema de PIMNL foi apresentada na Secção 2.1 sendo dada por (2.1.3). Estes problemas, devido à sua natureza combinatória, são considerados problemas **NP**-difíceis [GJ79]. Técnicas de optimização baseadas em gradientes apenas puderam ser aplicadas a formulações especiais, onde continuidade e convexidade tiveram de ser impostas, ou através da exploração de estruturas matemáticas particulares. Por outro lado, recentemente, abordagens base-

adas em algoritmos estocásticos foram utilizadas. Estas abordagens, também conhecidas como procura aleatória adaptativa, conseguiram com sucesso tratar problemas de PIMNL, principalmente na área da engenharia química [RRR83, Sal92, BS96].

Nos últimos anos, um grande número de trabalhos foram publicados sobre a aplicação de diversos Algoritmos Evolucionários (AEs) (Algoritmos Genéticos (AGs), Estratégias Evolutivas (EEs) e *Simulated Annealing*, etc.) na resolução de problemas de PIMNL em muitas aplicações de engenharia.

Nesta secção, sete problemas propostos por autores independentes são estudados utilizando AGs e EEs [CO99, CO01]. Estes problemas têm origem na área de engenharia química, e representam problemas de optimização não convexa difíceis, com variáveis contínuas e discretas. São feitas comparações com o algoritmo M-SIMPSA [CSFdA96a, CSFdA96b, CSFdAB97, Car98] baseado na combinação do método simplex de Nelder-Mead e *Simulated Annealing*.

4.1.1 Casos de Estudo

Os sete problemas da área da engenharia química aos quais foram aplicados os AEs são apresentados na Tabela 4.1. Nesta tabela, n e q são, respectivamente, o número de variáveis contínuas e inteiras do problema; m e p são, respectivamente, o número de restrições do tipo desigualdade e do tipo igualdade do problema. Os problemas #2 e #4, com restrições de igualdade, foram reformulados de forma a tornar estas restrições de igualdade em desigualdades (problemas #2′ e #4′).

Problema #1 Este problema tem uma restrição não linear e foi proposto por Kocis e Grossmann [KG88]; foi também resolvido por Floudas *et al.* [FAC89], Ryoo e Sahinidis [RS95] e Cardoso *et al.* [CSFdAB97].

$$\min f(x, y) = 2x + y$$

Problema	Variáveis				Restrições	
	n	q		$n+q$	m	p
	Reais	Inteiras	Binárias	Total	Desigualdade	Igualdade
#1	1	-	1	2	2	-
#2	2	-	1	3	1	1
#2'	1	-	1	2	1	-
#3	2	-	1	3	3	-
#4	7	-	2	9	4	6
#4'	4	-	1	5	6	-
#5	3	-	4	7	9	-
#6	3	2	-	5	3	-
#7	7	3	-	10	18	-

Tabela 4.1: Problemas de Programação Inteira Mista Não Linear

sujeito a

$$1.25 - x^2 - y \leq 0$$

$$x + y \leq 1.6$$

$$0 \leq x \leq 1.6$$

$$y \in \{0, 1\}$$

O óptimo global é $(x, y; f) = (0.5, 1; 2)$.

Problema #2 Este problema, com uma restrição não linear, foi proposto por Kocis e Grossmann [KG87] e também estudado por Salcedo [Sal92] e Cardoso *et al.* [CSFdAB97].

$$\min f(x_1, x_2, y) = -y + 2x_1 + x_2$$

sujeito a

$$x_1 - 2\exp(-x_2) = 0$$

$$-x_1 + x_2 + y \leq 0$$

$$0.5 \leq x_1 \leq 1.4$$

$$y \in \{0, 1\}$$

O óptimo global é $(x_1, x_2, y; f) = (1.375, 0.375, 1; 2.124)$.

Problema #2' O problema #2 também pode ser formulado sem a restrição de igualdade com o mesmo óptimo global.

$$\min f(x_1, y) = -y + 2x_1 - \ln(x_1/2)$$

sujeito a

$$-x_1 - \ln(x_1/2) + y \leq 0$$

$$0.5 \leq x_1 \leq 1.4$$

$$y \in \{0, 1\}$$

Problema #3 Este problema foi estudado em primeiro lugar por Floudas [Flo95] e apresenta uma restrição não linear. Foi também resolvido por Cardoso *et al.* [CSFdAB97].

$$\min f(x_{1,}x_{2,}y) = -0.7y + 5(x_1 - 0.5)^2 + 0.8$$

sujeito a

$$-\exp(x_1 - 0.2) - x_2 \leq 0$$

$$x_2 + 1.1y \leq -1$$

$$0.2 \leq x_1 \leq 1$$

$$-2.22554 \leq x_2 \leq -1$$

$$y \in \{0, 1\}$$

O óptimo global é $(x_1,x_2,y;f) = (0.94194,-2.1,1;1.07654)$.

Problema #4 Neste problema, retirado de Kocis e Grossmann [KG89], o objectivo é seleccionar um de dois reactores candidatos por forma a minimizar o custo de produção total. Este problema também foi resolvido por Diwekar *et al.* [DGR92], Diwekar e Rubin [DR93] e Cardoso *et al.* [CSFdAB97].

$$\min f(x,y_1,y_2,v_1,v_2) = 7.5y_1 + 5.5y_2 + 7v_1 + 6v_2 + 5x$$

sujeito a

$$y_1 + y_2 = 1$$

$$z_1 = 0.9\left[1-\exp(-0.5v_1)\right]x_1$$

$$z_2 = 0.8\left[1-\exp(-0.4v_2)\right]x_2$$

$$z_1 + z_2 = 10$$

$$x_1 + x_2 = x$$

$$z_1y_1 + z_2y_2 = 10$$

$$v_1 \leq 10y_1$$

$$v_2 \leq 10y_2$$

$$x_1 \leq 20y_1$$

$$x_2 \leq 20y_2$$

$$x_1,x_2,z_1,z_2,v_1,v_2 \geq 0$$

$$y_1,y_2 \in \{0,1\}$$

O óptimo global é $(x,y_1,y_2,v_1,v_2;f) = (13.362272,1,0,3.514237,0;99.245209)$.

Problema #4' O problema anterior, sem restrições de igualdade, pode ser formulado da seguinte forma:

$$\min f(y_1,v_1,v_2) = 7.5y_1 + 5.5(1-y_1) + 7v_1 + 6v_2 + 50\frac{1-y_1}{0.8[1-\exp(-0.4v_2)]} + 50\frac{y_1}{0.9[1-\exp(-0.5v_1)]}$$

sujeito a

$$0.9[1 - \exp(-0.5v_1)] - 2y_1 \leq 0$$

$$0.8[1 - \exp(-0.4v_2)] - 2(1 - y_1) \leq 0$$

$$v_1 \leq 10y_1$$

$$v_2 \leq 10(1 - y_1)$$

$$v_1, v_2 \geq 0$$

$$y_1 \in \{0, 1\}$$

Problema #5 Este problema foi estudado por Floudas *et al.* [FAC89], Yuan *et al.* [YZPD89], Salcedo [Sal92], Ryoo e Sahinidis [RS95] e Cardoso *et al.* [CSFdAB97] e apresenta várias restrições não lineares.

$$\begin{aligned}\min f(x_1, x_2, x_3, y_1, y_2, y_3, y_4) &= (y_1 - 1)^2 + (y_2 - 1)^2 \\ &+ (y_3 - 1)^2 - \ln(y_4 + 1) + (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2\end{aligned}$$

sujeito a

$$y_1 + y_2 + y_3 + x_1 + x_2 + x_3 \leq 5$$

$$y_3^2 + x_1^2 + x_2^2 + x_3^2 \leq 5.5$$

$$y_1 + x_1 \leq 1.2$$

$$y_2 + x_2 \leq 1.8$$

$$y_3 + x_3 \leq 2.5$$

$$y_4 + x_1 \leq 1.2$$

$$y_2^2 + x_2^2 \leq 1.64$$

$$y_3^2 + x_3^2 \leq 4.25$$

$$y_2^2 + x_3^2 \leq 4.64$$

$$x_1, x_2, x_3 \geq 0$$

$$y_1, y_2, y_3, y_4 \in \{0, 1\}$$

a_1	85.334407	a_5	80.51249	a_9	9.300961
a_2	0.0056858	a_6	0.0071317	a_{10}	0.0047026
a_3	0.0006262	a_7	0.0029955	a_{11}	0.0012547
a_4	0.0022053	a_8	0.0021813	a_{12}	0.0019085

Tabela 4.2: Dados do Problema #6

O óptimo global é $(x_1, x_2, x_3, y_1, y_2, y_3, y_4; f) = (0.2, 1.280624, 1.954483, 1, 0, 0, 1; 3.557463)$.

Problema #6 Este é um problema de maximização retirado de Wong [Won90] e foi também estudado por Cardoso *et al.* [CSFdAB97].

$$\max f(x_1, x_2, x_3, y_1, y_2) = -5.357854x_1^2 - 0.835689y_1x_3 - 37.29329y_1 + 40792.141$$

sujeito a

$$a_1 + a_2y_2x_3 + a_3y_1x_2 - a_4x_1x_3 \leq 92$$

$$a_5 + a_6y_2x_3 + a_7y_1y_2 + a_8x_1^2 - 90 \leq 20$$

$$a_9 + a_{10}x_1x_3 + a_{11}y_1x_1 + a_{12}x_1x_2 - 20 \leq 5$$

$$27 \leq x_1, x_2, x_3 \leq 45$$

$$y_1 \in \{78, \ldots, 102\}, \text{inteiro}$$

$$y_2 \in \{33, \ldots, 45\}, \text{inteiro}$$

onde a_1 a a_{12} são dados pela Tabela 4.2. O óptimo global é, para qualquer combinação de x_2, y_2, $(x_1, x_3, y_1; f) = (27, 27, 78; 32217.4)$.

Problema #7 Este é um problema de "*multi-product batch plant*" com M estágios de processamento em série, onde quantidades fixas de Q_i, a partir de N produtos, podem ser produzidas. O objectivo é encontrar para cada estágio j, o número de unidades paralelas N_j, bem como os tamanhos correspondentes V_j e, para cada produto i, os correspondentes tamanhos de lote B_i e tempos de ciclo T_{Li}. Os

dados são o horizonte de tempo H, os factores de tamanho S_{ij}, os tempos de processamento t_{ij} do produto i no estágio j, as quantidades a produzir requeridas Q_i e os coeficientes de custo α_j e β_j. Este problema, estudado por Grossmann e Sargent [GS79], Kocis e Grossmann [KG88], Salcedo [Sal92] e Cardoso *et al.* [CSFdAB97] tem a seguinte formulação matemática:

$$\min f = \sum_{j=1}^{M} \alpha_j N_j V_j^{\beta_j}$$

sujeito a

$$\sum_{i=1}^{N} \frac{Q_i T_{Li}}{B_i} \leq H$$

$$V_j \geq S_{ij} B_i$$

$$N_j T_{Li} \geq t_{ij}$$

$$1 \leq N_j \leq N_j^u$$

$$V_j^l \leq V_j \leq V_j^u$$

$$T_{Li}^l \leq T_{Li} \leq T_{Li}^u$$

$$B_j^l \leq B_j \leq B_j^u$$

$$N_j, \text{inteiro}$$

onde, para o problema específico considerado, $M = 3$, $N = 2$, $H = 6000$, $\alpha_j = 250$, $\beta_j = 0.6$, $N_j^u = 3$, $V_j^l = 250$ e $V_j^u = 2500$. Os valores de T_{Li}^l, T_{Li}^u, B_j^l e B_j^u são dados por:

$$T_{Li}^l = \max \frac{t_{ij}}{N_j^u}$$

$$T_{Li}^u = \max_j t_{ij}$$

$$B_i^l = \frac{Q_i}{H} T_{Li}$$

$$B_j^u = \min(Q_i, \min_j \frac{V_j^u}{S_{ij}})$$

A Tabela 4.3 apresenta os valores de S_{ij} e t_{ij}.

S_{ij}			t_{ij}		
2	3	4	8	20	8
4	6	3	16	4	4

Tabela 4.3: Dados do Problema #7

O óptimo global é $(N_1, N_2, N_3, V_1, V_2, V_3, B_1, B_2, T_{L1}, T_{L2}; f) = (1, 1, 1, 480, 720, 960, 240, 120, 20, 16; 38499.8)$.

4.1.2 Descrição do Algoritmo Genético

Nesta secção descrevem-se as principais características do AG utilizado nas experiências computacionais. Uma vez que o funcionamento dos AGs foi apresentado com detalhe na Secção 3.2 do capítulo anterior, apenas se vão indicar os aspectos relacionadas com a aplicação específica de AGs a problemas de MINLP.

Representação das Soluções

A implementação de um AG para resolver problemas de PIMNL requer a representação das potenciais soluções. Um problema de PIMNL possui variáveis de decisão contínuas e discretas (binárias e/ou inteiras). Logo, cada potencial solução, um vector de variáveis contínuas e inteiras, é representado por uma sequência de dígitos binários, o cromossoma.

O tamanho do cromossoma depende do número de variáveis de decisão do problema considerado. As variáveis, codificadas utilizando uma representação binária, foram definidas da seguinte forma:

- as n variáveis contínuas são representadas por um determinado número de dígitos binários que define a resolução (em geral, utilizou-se uma representação binária de 32 dígitos binários);

- cada variável inteira (definida num intervalo de valores discretos) é representada pelo menor número de dígitos binários que possibilita a representação de todos os valores, e, se necessário, são introduzidas restrições para evitar valores não admissíveis para a variável;

- as variáveis binárias são naturalmente representadas em código binário (um dígito binário por variável).

De acordo com este esquema de representação, o tamanho dos cromossomas depende do número de variáveis contínuas e inteiras do problema a resolver.

Parâmetros do AG

A Tabela 4.4 apresenta os valores dos parâmetros do AG utilizados. Estes valores foram fixados após alguma experimentação. Cada problema foi resolvido 10 vezes. Em qualquer uma das execuções utilizou-se a selecção uniforme estocástica [Bak87], a recombinação com dois pontos (cruzamento duplo) com probabilidade de 0.7, e a mutação uniforme com probabilidade de 0.001. Em todas as execuções, a população inicial consistiu em 250 cromossomas gerados aleatoriamente. As soluções foram codificadas utilizando uma sequência de dígitos binários em código *Gray*.

Tratamento das Restrições

Numa primeira fase, todas as restrições foram tratadas utilizando um esquema de penalização (ver a equação (3.2.12)). Os seguintes valores do coeficiente de penalização foram considerados: 10, 10^3 e 10^5. Numa segunda fase, as desigualdades foram tratadas de acordo com o esquema proposto por Deb [Deb00] (ver a equação (3.2.13)). Este esquema não requer nenhum coeficiente de penalização. Numa dada geração, o parâmetro $f_{\max}$ foi estimado pelo valor da função objectivo do pior cromossoma na população.

Parâmetro	Valor
Número de experiências	10
Tamanho da população	250
Tamanho do cromossoma	Variável
Probabilidade de recombinação	0.7
Probabilidade de mutação	0.001
Coeficiente de penalização	10, 10^3, 10^5
Mecanismo de codificação	Código *Gray*

Tabela 4.4: Parâmetros do Algoritmo Genético

Critério de Paragem

O critério de paragem adoptado consistiu em terminar a procura quando uma das seguintes condições fosse verificada:

- o número máximo de gerações fosse atingido (1000 gerações);
- um determinado número de gerações fosse atingido sem que ocorresse uma melhoria nos valores da função objectivo (50 gerações);
- todos os indivíduos da população convergissem para um único ponto, i.e., quando toda a população convergisse para o mesmo cromossoma.

4.1.3 Descrição das Estratégias Evolutivas

Nesta secção descrevem-se as principais características das EEs utilizadas, salientando-se, em especial, os aspectos relacionados com a sua aplicação a problemas de MINLP. Consideraram-se três tipos de EEs: EE-(1+1), EE-(10+100) e EE-(10,100). Em qualquer dos casos, nenhum tipo de recombinação foi utilizado.

Parâmetro	Valor
Número de experiências	10
Aproximação inicial	Variável
Valor inicial de σ	Variável
Limite inferior de σ (absoluto)	10^{-5}
Limite inferior de σ (relativo)	10^{-5}
Factor de actualização de σ	0.85

Tabela 4.5: Parâmetros das Estratégias Evolutivas

Representação das Soluções

As EEs utilizam directamente os valores reais das variáveis de decisão. Logo, para as variáveis contínuas nenhum procedimento especial é requerido. As variáveis binárias foram representadas por dígitos binários e as inteiras por valores inteiros. Por isso, na aplicação do operador de mutação às variáveis binárias e inteiras foi considerada uma distribuição de probabilidade discreta (distribuição Binomial).

Parâmetros das EEs

A Tabela 4.5 apresenta os parâmetros utilizados nas estratégias evolutivas. Tal como anteriormente, cada problema foi resolvido 10 vezes. Os valores iniciais para os desvios padrão σ_i (com $i = 1, \ldots, n+q$) foram os sugeridos para as EEs em optimização uni-objectivo; i.e., os valores iniciais para σ_i foram dados pela equação (3.1.1) com $\Delta x_k = (\overline{x_k} - \underline{x_k})/2$ (onde $\overline{x_k}$ e $\underline{x_k}$ são os limites superior e inferior da variável x_k com $k = 1, \ldots, n$) e $\Delta y_l = (\overline{y_l} - \underline{y_l})/2$ (onde $\overline{y_l}$ e $\underline{y_l}$ são os limites superior e inferior da variável y_l com $l = 1, \ldots, q$). No caso da EE-(1+1) utilizou-se a "Regra de 1/5 Sucessos" (com $c_{dec} = 0.85$ e $c_{inc} = 1/0.85$) para actualizar os desvios padrão σ_i.

Tratamento de Restrições

Todas restrições foram tratadas utilizando um mecanismo de eliminação (todos os pontos não admissíveis são eliminados). Dependendo do problema, uma aproximação inicial ao óptimo global foi considerada. Esta aproximação inicial permitiu a geração aleatória da população inicial. Um mecanismo simples foi utilizado para garantir que todos os pontos da população inicial eram admissíveis.

Critério de Paragem

O critério de paragem adoptado para a EE-(1+1) consistiu em terminar a execução quando uma das seguintes condições fosse verificada:

- o número máximo de gerações fosse atingida (1000 gerações);
- $\left|f^{(k+\Delta k)} - f^{(k)}\right| < 10^{-5}$ com $\Delta k = 20$;
- $\frac{\left|f^{(k+\Delta k)} - f^{(k)}\right|}{\left|f^{(k+\Delta k)}\right|} < 10^{-5}$ com $\Delta k = 20$.

Para as EE-$(10 + 100)$ e EE-$(10, 100)$, o critério de paragem adoptado foi terminar a execução quando uma das seguintes condições fosse verificada:

- o número máximo de gerações fosse atingida (1000 gerações);
- $\left|f_{\max}^{(k)} - f_{\min}^{(k)}\right| < 10^{-5}$;
- $\frac{\left|f_{\max}^{(k)} - f_{\min}^{(k)}\right|}{\left|\bar{f}^{(k)}\right|} < 10^{-5}$.

4.1.4 Resultados Computacionais

As Figuras 4.1 e 4.2 resumem os resultados obtidos quando o AG foi utilizado. Nestas experiências, diferentes valores do coeficiente de penalização das restrições (R igual a 10, 10^3 e 10^5), e o esquema de Deb foram considerados. A Figura 4.1 apresenta a percentagem

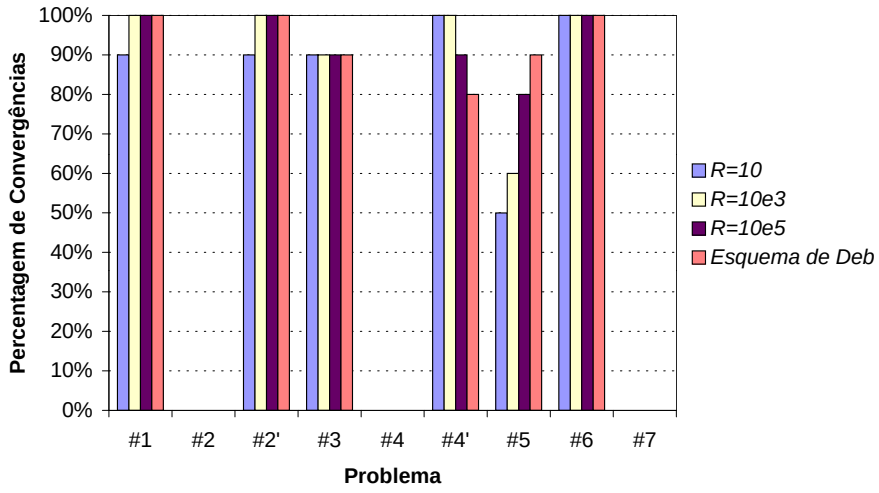

Figura 4.1: Resultados obtidos pelo Algoritmo Genético (% Convergências)

de convergências para o óptimo global e a Figura 4.2 mostra o número médio de cálculos da função objectivo necessários para atingir a convergência.

As Figuras 4.3 e 4.4 resumem os resultados obtidos quando as EEs foram consideradas. Nestas experiências, três EEs foram aplicadas, a EE-$(1 + 1)$, a EE-$(10 + 100)$ e a EE-$(10, 100)$. Como anteriormente, a Figura 4.3 apresenta a percentagem de convergências para o óptimo global e a Figura 4.4 mostra o número médio de cálculos da função objectivo necessários para atingir a convergência. A Tabela 4.6 apresenta a comparação, em termos do número de cálculos da função objectivo e a proporção de convergências para o óptimo global, entre o AG (com dois esquemas de penalização), a EE-$(10 + 100)$ e o algoritmo M-SIMPSA. Nesta tabela, $\overline{\#F}$ e $C\%$ representam, respectivamente, o número médio de cálculos da função objectivo nas 10 execuções, e a percentagem de convergências para o óptimo global. Os resultados apresentados são os melhores obtidos para cada um dos problemas.

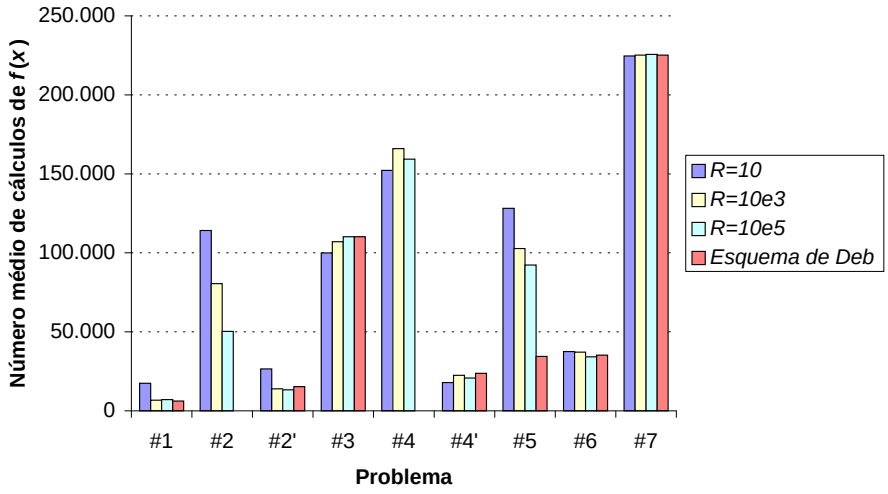

Figura 4.2: Número de Avaliações da Função Objectivo requeridas pelo Algoritmo Genético

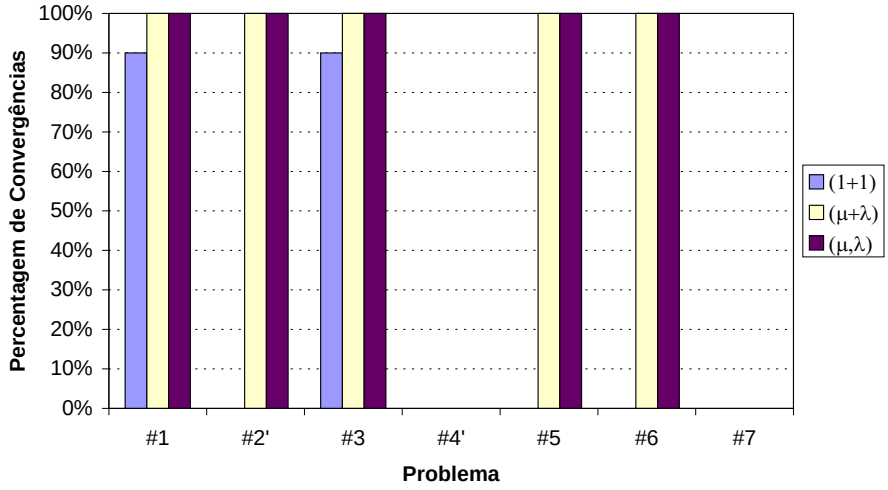

Figura 4.3: Resultados obtidos pelas Estratégias Evolutivas (% Convergências)

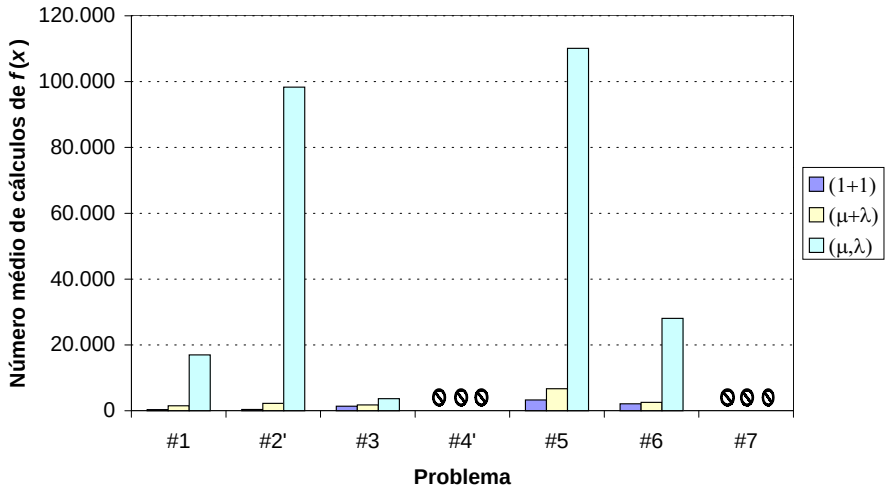

Figura 4.4: Número de Avaliações da Função Objectivo requeridas pelas Estratégias Evolutivas

	AG-$R = 10^3$		AG-Esq. Deb		EE-(10+100)		M-SIMPSA		M-SIMPSA-pen.	
Prob. #	$\overline{\#F}$	$C\%$	$\overline{\#F}$	$C\%$	$\overline{\#F}$	$C\%$	$\overline{\#F}$	$C\%$	$\overline{\#F}$	$C\%$
1	6787	**100**	6191	**100**	**1518**	**100**	607	99	16282	**100**
2'	13939	**100**	15298	**100**	**2255**	**100**	10582	83	14440	**100**
3	107046	90	110233	90	**1749**	**100**	-	0	38042	**100**
4'	22489	**100**	23730	80	- (*)	0	**14738**	**100**	42295	**100**
5	102778	60	34410	90	**6710**	**100**	22309	60	63751	97
6	37167	**100**	35255	**100**	**2536**	**100**	27410	87	33956	95
7	225176	0	225173	0	- (*)	0	-	0	**257536**	**97**

* - execução abortada

Tabela 4.6: Algoritmo Evolucionários versus M-SIMPSA

4.1.5 Discussão dos Resultados

Deve-se destacar que o objectivo deste trabalho não foi encontrar o "melhor" algoritmo para todas as instâncias de problemas, mas comparar os diferentes algoritmos para tentar encontrar classes de problemas que podem ser mais adequadas para certos algoritmos do que outros.

Uma das principais desvantagens dos AEs reside no tempo computacional requerido. Por outro lado, estes requerem apenas os valores da função objectivo para guiar a procura que é baseada numa população de pontos. Por este motivo, os AEs conseguem lidar com problemas não convexos com uma probabilidade reduzida de convergirem para um óptimo local. Os resultados obtidos pelo AG e pelas EEs foram comparados com os resultados previamente publicados do M-SIMPSA [CSFdAB97], um algoritmo baseado em *Simulated Annealing*.

Os resultados mostram claramente a dificuldade de manuseamento de restrições do tipo igualdade (Problemas #2 e #4); no entanto, quando estes problemas são reformulados sem este tipo de restrições, os algoritmos exibiram uma razão de convergência elevada.

O método proposto por Deb [Deb00] para lidar com restrições é claramente superior ao esquema de penalização usual, uma vez que não requer nenhum parâmetro de penalização. A proposta de Cardoso *et al.* [CSFdAB97] para o algoritmo M-SIMPSA pode gerar pontos não admissíveis com valor da função de desempenho superior a pontos admissíveis. As EEs excluem todos os pontos não admissíveis, o que corresponde a ter um valor infinito para o coeficiente de penalização. Todos os algoritmos apresentaram grandes dificuldades com o problema #7, o que não é surpreendente, uma vez que o problema é fortemente restrito; o óptimo global corresponde a um ponto onde uma pequena variação em qualquer uma das variáveis contínuas produz não admissibilidade.

As EEs apresentaram dificuldades com problemas fortemente restritos, mas, em geral,

são mais eficientes em termos do número de cálculos da função objectivo. Em resumo, a resolução de problemas PIMNL com AEs é uma abordagem válida em problemas não convexos onde o tempo de computação não seja de primeira importância.

4.2 Optimização de Placas Laminadas de Materiais Isotrópicos

Nesta secção é descrita a aplicação de um Algoritmo Genético (AG) ao problema de optimização de placas laminadas de materiais isotrópicos [FFJ+98, COIL99]. O problema de optimização é formulado como um problema de programação inteira com restrições.

Os problemas de optimização estrutural dos tamanhos e materiais de placas são, em geral, formulados como problemas contínuos não lineares [HN98, BS93, Ben95]. As soluções numéricas são obtidas por algoritmos de gradiente convencionais que requerem derivadas bem como propriedades de convexidade, convergindo, em geral, para soluções óptimas locais. No entanto, a modelação de problemas do mundo real impõe (por razões quer de ordem natural quer tecnológica) a consideração de variáveis inteiras, como sejam os diferentes materiais e/ou diferentes espessuras, escolhidas de um conjunto de valores discretos. No entanto, isto conduz a problemas inteiros mistos não lineares, que não possuem as propriedades de diferenciabilidade e convexidade.

O problema estrutural considerado está relacionado com a rigidez de uma placa elástica linearmente laminada. Neste problema, cada lâmina é feita de um único material isotrópico e homogéneo, e as variáveis de decisão são a espessura, o módulo de *Young* e o coeficiente de *Poisson*. Este problema estrutural pode ser visto como um problema combinatório em relação às variáveis relacionadas com os materiais, o módulo de *Young* e o coeficiente de *Poisson*, que assumem valores discretos. No entanto, a espessura pode tomar qualquer valor de um intervalo contínuo e são impostos limites superiores na massa, preço e espessura

global da placa.

O modelo estrutural considerado pode ser formulado como um problema de optimização com um único objectivo, a complacência da estrutura e algumas restrições associadas à espessura, preço e massa da estrutura. A aplicação de um AG a este problema de optimização impõe que estas restrições sejam parte de uma função de penalização.

4.2.1 Formulação do Problema

Foi considerada uma placa elástica, feita de várias lâminas, que são simétricas em relação ao plano médio da placa definida pelo conjunto $\Omega \subset \Re^2$. Devido a esta simetria, o número total de camadas compondo a placa é igual a $2k$, onde k é um inteiro maior do que 1. Cada lâmina, com espessura t_i, é associada exactamente a um material m_j.

São definidos os vectores

$$\mathbf{t} = (t_1, \ldots, t_k) \qquad \text{e} \qquad \mathbf{m} = (m_1, \ldots, m_p), \tag{4.2.1}$$

que correspondem aos vectores de espessura e materiais, respectivamente. O problema de optimização estrutural consiste em encontrar a melhor distribuição de materiais e espessuras que produza a placa mais rígida. A formulação discreta deste modelo que é obtida pelo método de elementos finitos, corresponde ao seguinte problema de minimização:

$$\left[\begin{array}{l} \min\limits_{(\mathbf{t},\mathbf{m})\in C} \frac{1}{2}\mathbf{f}^T\mathbf{u} \\ \text{sujeito a} \quad \mathbf{K}(\mathbf{t},\mathbf{m})\mathbf{u} = \mathbf{f} \end{array} \right. . \tag{4.2.2}$$

As variáveis deste problema de minimização (4.2.2), i.e., as variáveis de decisão, os vectores $\mathbf{u}$ e $\mathbf{f}$, a matriz $\mathbf{K}$ e o conjunto de restrições C, são discutidas em seguida.

Os vectores $\mathbf{t} \in \Re^k$ e $\mathbf{m} \in \Re^p$ contêm as variáveis de decisão t_i e m_j associadas às espessuras e materiais da placa. Para definir a espessura t_i de cada lâmina $i = 1, 2, \ldots, k$, seja h_i a distância da superfície média Ω à face superior da lâmina i (Figura 4.5). Por isso,

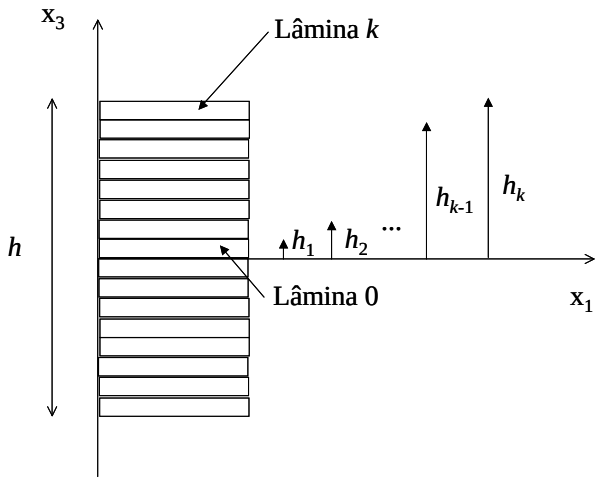

Figura 4.5: Estrutura de uma Placa Laminada com Materiais Isotrópicos

$t_i = h_i - h_{i-1}$ é a espessura da lâmina i e

$$2\sum_{i=1}^{k} t_i = 2\sum_{i=1}^{k}(h_i - h_{i-1})$$

é a espessura total da placa ($h_0 = 0$).

Cada material m_j, com $j = 1, 2, \ldots, p$, do conjunto de materiais admissível, é definido pelo par (E_j, ν_j), onde E_j é o módulo de *Young*, e ν_j é coeficiente de *Poisson*, i.e., $m_j = (E_j, \nu_j)$. Uma vez que não são permitidos materiais repetidos, e cada lâmina i deve ser de um único material, (E_j, ν_j), escolhido de um conjunto de p materiais, o material m_i na lâmina i é dado por:

$$m_i = \sum_{j=1}^{p} x_{ij}(E_j, \nu_j), \quad \text{para} \quad i \in \{1, 2, \ldots, k\}, \tag{4.2.3}$$

onde

$$\sum_{j=1}^{p} x_{ij} = 1, \quad \text{para} \quad i = 1, \ldots, k, \qquad \text{e} \qquad \sum_{i=1}^{k} x_{ij} = 1, \quad \text{para} \quad j = 1, \ldots, p, \tag{4.2.4}$$

e

$$x_{ij} \in \{0, 1\}, \quad \text{para} \quad i = 1, \ldots, k, \quad j = 1, \ldots, p. \tag{4.2.5}$$

O vector $\mathbf{u} \in \Re^q$ representa a aproximação ao deslocamento vertical da placa em determinados pontos da superfície média (q é o número de graus de liberdade global da malha de elementos finitos).

O vector $\mathbf{f} \in \Re^q$ é a força vertical que actua na placa.

A matriz quadrada $\mathbf{K}(\mathbf{t}, \mathbf{m})$ é a matriz de rigidez, e é dada por

$$\mathbf{K}(\mathbf{t}, \mathbf{m}) = \sum_{i=1}^{k} K_i(x_{ij}, t_i), \tag{4.2.6}$$

onde para cada x_{ij} e t_i

$$\mathbf{K}_i(x_{ij}, t_i) = \int_\Omega \mathbf{B}^T \mathbf{D}_i(x_{ij}, t_i)\mathbf{B} \tag{4.2.7}$$

é uma matriz semi-definida positiva de ordem q. As definições das matrizes $\mathbf{B}$ e $\mathbf{D}_i$ são apresentadas em seguida:

i) $\mathbf{B}$ é uma matriz $3 \times q$ das derivadas de segunda ordem das funções da forma global N_1, N_2, ..., N_q, independente de (x_{ij}, t_i), e é dada por

$$\mathbf{B} = \begin{pmatrix} N_{1,11} & N_{2,11} & \cdots & N_{q,11} \\ N_{1,22} & N_{2,22} & \cdots & N_{q,22} \\ 2N_{1,12} & 2N_{2,12} & \cdots & 2N_{q,12} \end{pmatrix}, \tag{4.2.8}$$

onde N_1, N_2, ..., N_q são funções definidas em Ω e $N_{l,rs}$ denota as segundas derivadas de N_l, para $l = 1, \ldots, q$, $r = 1, 2$ e $s = 1, 2$.

ii) A matriz $\mathbf{D}_i(x_{ij}, t_i)$ é a matriz de elasticidade de ordem 3, associada à lâmina i e satisfaz

$$\left[\mathbf{D}_i(x_{ij}, t_i) = \frac{2}{3}(h_i^3 - h_{i-1}^3) \frac{\sum_{j=1}^{p} x_{ij} E_j}{1 - (\sum_{j=1}^{p} x_{ij} \upsilon_j)^2} \right. \left(\begin{array}{ccc} 1 & \sum_{j=1}^{p} x_{ij} \upsilon_j & 0 \\ \sum_{j=1}^{p} x_{ij} \upsilon_j & 1 & 0 \\ 0 & 0 & \frac{1 - \sum_{j=1}^{p} x_{ij} \upsilon_j}{2} \end{array} \right) \tag{4.2.9}$$

O conjunto C é o conjunto de espessuras e materiais admissíveis, e inclui algumas restrições, tais como os limites superiores e inferiores das espessuras, bem como os limites superiores e inferiores para a espessura global, preço e massa da placa. Em seguida, estas restrições são descritas. No que diz respeito ao vector de espessuras $\mathbf{t}$, as restrições são definidas pelas expressões

$$\begin{cases} t_i^{\min} \leq t_i \leq t_i^{\max}, \quad i = 1, \ldots, k, \\ t_i^{\min} \quad \text{e} \quad t_i^{\max} \ \text{conhecido para cada } i \\ \sum_{i=1}^{k} t_i^{\min} \leq \sum_{i=1}^{k} t_i \leq T_1 < \sum_{i=1}^{k} t_i^{\max} \end{cases} \tag{4.2.10}$$

onde T_1 é uma restrição de espessura total. Para além disso, a massa total deve ser menor ou igual do que uma dada constante M_1. Esta restrição pode ser escrita da seguinte forma

$$S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij} \ \rho_j \leq M_1 \tag{4.2.11}$$

onde S é a área da superfície média da placa, ρ_j é a densidade de massa do material j e x_{ij} são definidas em (4.2.5). Finalmente, o preço total deve ser menor ou igual que um determinado preço P_1, por isso

$$S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij} \ \rho_j \ p_j \leq P_1 \tag{4.2.12}$$

onde p_j é o preço relativo do material j, S e x_{ij} são os mesmos como em (4.2.11).

Em conclusão, o problema de optimização da placa pode ser formulado como o seguinte

problema de programação inteira mista,

$$\text{Minimizar} \quad g_1(\mathbf{t},\mathbf{x}) = \frac{1}{2}\mathbf{f}^T\mathbf{u} \tag{4.2.13}$$

$$\text{sujeito a} \quad \mathbf{K}(\mathbf{t},\mathbf{m})\mathbf{u} = \mathbf{f} \tag{4.2.14}$$

$$t_i^{\min} \le t_i \le t_i^{\max}, \quad i = 1,\ldots,k \tag{4.2.15}$$

$$\sum_{j=1}^{p} x_{ij} = 1, \quad i = 1,\ldots,k \tag{4.2.16}$$

$$\sum_{i=1}^{k} x_{ij} = 1, \quad j = 1,\ldots,p \tag{4.2.17}$$

$$x_{ij} \in \{0,1\}, \quad i = 1,\ldots,k, \quad j = 1,\ldots,p \tag{4.2.18}$$

$$g_2(\mathbf{t},\mathbf{x}) \le T_1 \tag{4.2.19}$$

$$g_3(\mathbf{t},\mathbf{x}) \le M_1 \tag{4.2.20}$$

$$g_4(\mathbf{t},\mathbf{x}) \le P_1 \tag{4.2.21}$$

onde $\mathbf{t} = (t_i) \in \Re^k$, $\mathbf{m} = (m_j) \in \Re^p$ satisfaz (4.2.3), $\mathbf{x} = (x_{ij}) \in \Re^{k\times p}$, $\mathbf{K}(\mathbf{t},\mathbf{m})$ é uma matriz $q \times q$ dada por (4.2.7), $\mathbf{f} \in \Re^q$, e g_2, g_3 e g_4 são funções definidas pelas seguintes expressões

$$g_2(\mathbf{t},\mathbf{x}) = \sum_{i=1}^{k} t_i \tag{4.2.22}$$

$$g_3(\mathbf{t},\mathbf{x}) = S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij}\ \rho_j \tag{4.2.23}$$

$$g_4(\mathbf{t},\mathbf{x}) = S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij}\ \rho_j\ p_j. \tag{4.2.24}$$

Neste formulação matemática, as restrições (4.2.15-4.2.16-4.2.17-4.2.18) devem ser distinguidas, uma vez que podem ser facilmente satisfeitas fixando uma única variável x_{ij} a 1, todas as restantes variáveis a zero e forçando cada uma das variáveis t_i a pertencer ao intervalo com os correspondentes limites superiores e inferiores, $t_i^{\min}$ e $t_i^{\max}$, respectivamente. Além disso, o valor da função objectivo para uma tal solução é fácil de calcular,

resolvendo-se o sistema linear (4.2.14) depois de fixar previamente os valores das variáveis $\mathbf{t}$ e $\mathbf{m}$ de acordo com (4.2.7).

Um algoritmo baseado apenas nos valores da função objectivo e, incorporando algumas técnicas que permitam lidar com as restrições (4.2.19-4.2.20-4.2.21), parece ser recomendável para resolver este problema de programação inteira. Por isso, tal como é descrito na secção seguinte, foi aplicado um AG utilizando um esquema de penalização para lidar com as restrições.

4.2.2 Descrição do Algoritmo Genético

Codificação Genética – Representação do Cromossoma

A implementação de um AG para o problema de optimização do projecto de placas laminadas requer a consideração de uma abordagem a dois níveis: a solução do problema estrutural (que é a solução da equação $\mathbf{K}(\mathbf{t}, \mathbf{m})\mathbf{u} = \mathbf{f}$) e o problema de minimização (4.2.2). Para um dado conjunto de materiais e respectivas espessuras, a solução do problema estrutural pode ser conseguida através de um programa de elementos finitos. Cada combinação de materiais m_i e espessuras t_i, pode ser visto como um ponto do espaço de procura para o AG. Logo, a solução estrutural é a medida do desempenho associada a cada cromossoma, um conjunto de materiais e espessuras. Cada cromossoma é avaliado pelo programa de elementos finitos, permitindo a ordenação em termos do desempenho de todos os cromossomas presentes na população numa dada geração.

Numa primeira abordagem, uma placa rectangular simétrica é considerada, com um número máximo de 10 camadas, e sem nenhum material repetido. No entanto, devido à simetria, é apenas necessário considerar 5 camadas, i.e., $k = 5$. Logo, uma placa é descrita por 5 variáveis contínuas, respeitantes às espessuras de cada camada; estas variáveis são codificadas utilizando para cada uma, uma representação binária de 4 bits, totalizando 20

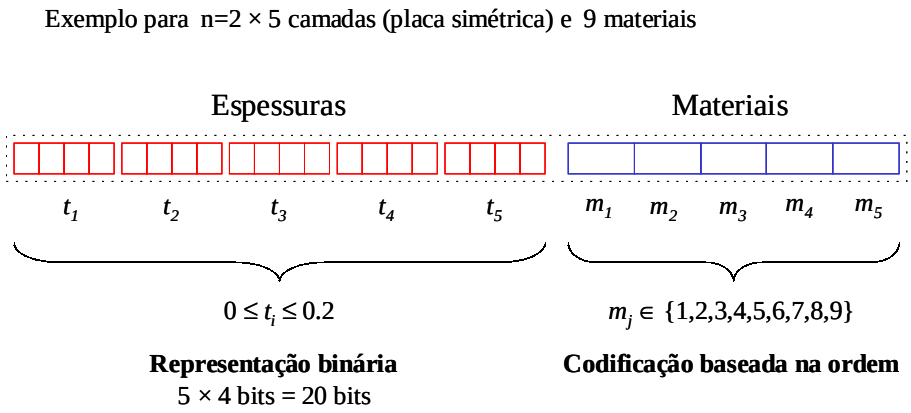

Figura 4.6: Codificação

bits. Uma vez que cada camada é feita de um único material, e não existem materiais repetidos, 5 variáveis inteiras ordinais são usadas para representar a ordem dos materiais na placa. É possível existir uma camada com espessura nula, o que corresponde a uma placa com um número total de apenas 8 camadas. Assim, cada cromossoma tem um comprimento total de 25 posições (alelos), onde 20 alelos representam as variáveis contínuas e 5 as variáveis inteiras ordinais (Figura 4.6).

Operadores Genéticos

A implementação dos diversos operadores genéticos não criou qualquer dificuldade especial, uma vez que cada um foi aplicado de forma independente aos dois tipos de variáveis na representação do cromossoma. Para cada tipo de variável (contínua t_i ou inteira m_j) foram desenvolvidos operadores genéticos distintos. Às variáveis contínuas t_i aplicou-se a recombinação com dois pontos (cruzamento duplo) e a mutação uniforme com diferentes probabilidades. Para as variáveis inteiras ordinais implementou-se uma recombinação ordinal com máscara e uma mutação com sub-lista (Figuras 4.7 e 4.8) [Dav91]. Estes operadores genéticos evitam a geração de soluções não admissíveis, i.e., é impossível ge-

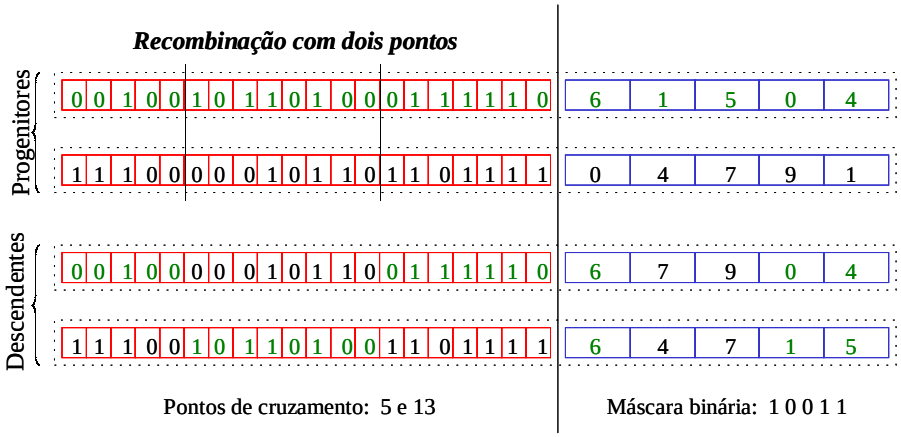

Figura 4.7: Recombinação

rar soluções com materiais repetidos. Tal como a recombinação com dois pontos para representações binárias, a recombinação ordinal preserva parte do primeiro progenitor e incorpora informação do segundo progenitor, apesar da sua implementação ser um pouco mais complexa. Em seguida, o processo de geração do primeiro descendente a partir de dois progenitores (para o segundo descendente é feito o processo complementar) é apresentado:

1. é criada aleatoriamente uma máscara de dígitos binários;

2. se, numa determinada posição, a máscara de dígitos binários tem um 1 então o gene é copiado a partir do primeiro progenitor;

3. as posições restantes são preenchidas com os genes do segundo progenitor que sejam diferentes dos genes previamente copiados do primeiro progenitor (de acordo com a ordem em que aparecem).

A mutação com sub-lista selecciona aleatoriamente uma sub-lista do progenitor, sendo depois a ordem dos elementos da sub-lista invertida.

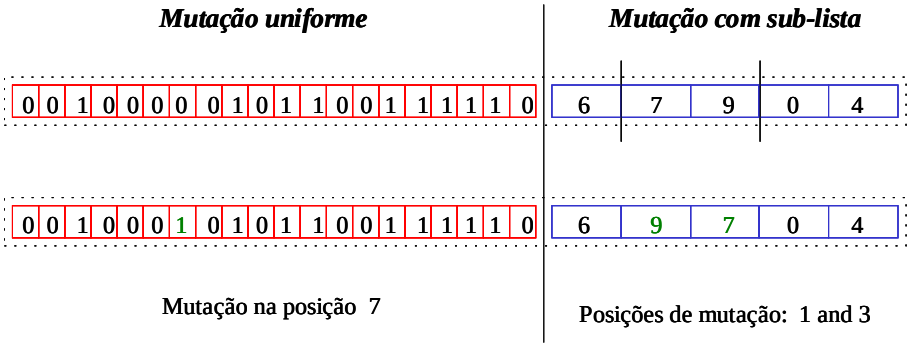

Figura 4.8: Mutação

Função Objectivo e Restrições

São consideradas diversas restrições como as indicadas em (4.2.15)-(4.2.21), nomeadamente a espessura máxima e mínima de cada camada (4.2.15), a espessura total da placa (4.2.19), a massa total da placa (4.2.20) e o preço total da placa (4.2.21). Um número limitado de materiais, satisfazendo as restrições (4.2.3-4.2.4-4.2.5), correspondendo a (4.2.16-4.2.17-4.2.18), é considerado. Na abordagem baseada em penalização, as restrições difíceis (4.2.19-4.2.20-4.2.21) que restringem o espaço de procura são incluídas na função objectivo através de um termo de penalização levando à seguinte função

$$G(\mathbf{t},\mathbf{x},\mathbf{u}) = \frac{1}{2}\mathbf{f}^T\mathbf{u} + R\sum_{i=1}^{3}\Big[C_i(\mathbf{t},\mathbf{x})\Big]^2 \tag{4.2.25}$$

onde R é o coeficiente de penalização e as funções $C_i(\mathbf{t},\mathbf{x})$, $i = 1,2,3$, são definidas por

$$\left[\begin{array}{l} C_1(\mathbf{t},\mathbf{x}) = \min\{0,\ T_1 - g_2(\mathbf{t},\mathbf{x})\} \\ C_2(\mathbf{t},\mathbf{x}) = \min\{0,\ M_1 - g_3(\mathbf{t},\mathbf{x})\} \\ C_3(\mathbf{t},\mathbf{x}) = \min\{0,\ P_1 - g_4(\mathbf{t},\mathbf{x})\} \end{array}\right. \tag{4.2.26}$$

com g_i sendo as funções dadas por (4.2.22-4.2.23-4.2.24).

Cada material é caracterizado pelo seu módulo de *Young*, coeficiente de *Poisson*, massa específica e preço relativo. Para um dado cromossoma, especificando um conjunto de materiais com as respectivas espessuras, o pacote de *software* CALFEM [LU97] é utilizado

para avaliar a solução $\mathbf{u}$ da equação $\mathbf{K}(\mathbf{t}, \mathbf{m})\mathbf{u} = \mathbf{f}$. O elemento finito de placas "*platre*" do código CALFEM (cf. p.5.7-1, [LU97]) é utilizado para calcular a matriz de rigidez $\mathbf{K}_i$ de cada lâmina i. Este elemento finito "*platre*" é um rectângulo, aproximado para um material isotrópico elástico linear, com doze graus de liberdade (os quatro deslocamentos e as duas derivadas do deslocamento vertical, nos vértices do rectângulo). A solução $\mathbf{u}$ é então incorporada juntamente com as restrições de espessura, massa e preço (4.2.19), (4.2.20) e (4.2.21), na função objectivo definida por (4.2.25).

4.2.3 Resultados Computacionais

A lista de materiais considerada é apresentada na Tabela 4.7. Foram feitas diversas experiências com o objectivo de testar se o modelo poderia gerar resultados com significado físico (listadas na Tabela 4.8). Em todas as experiências, a restrição na espessura máxima de cada camada ($0.2mm$) foi considerada. O problema #0 representa a experiência sem nenhuma restrição com excepção da espessura máxima de cada camada. Os problemas #1 a #6 apresentam a conjunção de diferentes restrições da massa total, espessura total e preço relativo.

Diversas experiências preliminares foram feitas para se ajustar os parâmetros do AG. Estas experiências incluíram a variação das probabilidades de recombinação e mutação. A probabilidade de recombinação foi escolhida de entre os valores 0.6, 0.7 e 0.8, e a probabilidade de mutação de entre os valores 0.0005, 0.001 e 0.005. Após esta fase, os parâmetros foram fixados como se apresentam na Tabela 4.9. As experiências foram executadas num computador pessoal com um processador Pentium III. Deve ser notado que 95 % do tempo de execução total é dedicado à execução da sub-rotina CALFEM para encontrar o vector de deslocamento $\mathbf{u}$.

Para os problemas considerados nas experiências, a função objectivo tem um custo computacional elevado. Além disso, para esta formulação do problema com um objectivo, ao

Material	Módulo de *Young* (Kg/cm^2)	Coeficiente de *Poisson*	Massa Específica (Kg/cm^3)	Preço Relativo (Preço/Kg)
Aço (**S**)	21×10^5	0.28	7.8×10^{-3}	1
Ferro (**I**)	20×10^5	0.28	7.4×10^{-3}	0.65
Cobre (**C**)	11×10^5	0.34	8.9×10^{-3}	6.25
Bronze (**B**)	10×10^5	0.31	8.25×10^{-3}	4.5
Alumínio (**A**)	7×10^5	0.34	2.6×10^{-3}	5
Vidro (**G**)	5.5×10^5	0.25	2.7×10^{-3}	3.75
Níquel (**N**)	20.7×10^5	0.29	7.75×10^{-3}	6.25
Chumbo (**L**)	1.8×10^5	0.44	11.34×10^{-3}	3
Borracha (**R**)	0.037×10^5	0.485	1.8×10^{-3}	5

Tabela 4.7: Lista de Materiais Isotrópicos

Problema	Espessura (mm)	Espessura Total (mm)	Massa Total (Kg)	Preço Relativo Total
#0	≤ 0.2	-	-	-
#1	≤ 0.2	≤ 0.8	≤ 2	≤ 10
#2	≤ 0.2	≤ 0.8	≤ 2	≤ 5
#3	≤ 0.2	≤ 0.8	≤ 1	≤ 5
#4	≤ 0.2	≤ 0.5	≤ 2	≤ 5
#5	≤ 0.2	≤ 0.8	≤ 0.8	≤ 5
#6	≤ 0.2	≤ 0.8	≤ 2	≤ 1

Tabela 4.8: Instâncias do Problema

Parâmetro	Valor
Número de experiências	4
Número máximo de gerações	1000
Tamanho da população	100
Probabilidade de recombinação (em dois pontos)	0.7
Probabilidade de mutação (uniforme)	0.001
Probabilidade de recombinação (uniforme baseada na ordem)	0.7
Probabilidade de mutação (sub-lista)	0.001
Coeficiente de penalização	1000

Tabela 4.9: Parâmetros do Algoritmo Genético

longo da procura, as mesmas soluções são avaliadas diversas vezes, especialmente, quando o algoritmo está prestes a convergir. Portanto, um esquema minimizando a repetição dos mesmos cálculos parece ser muito útil para este tipo de problemas. O esquema implementado mantém uma lista contendo as soluções e os respectivos valores da função objectivo mais vezes utilizados ao longo das sucessivas gerações. Este esquema permitiu reduzir, em média, 15% do número total de cálculos da função objectivo (os tempos de execução são também reduzidos proporcionalmente).

A Tabela 4.10 apresenta as melhores soluções obtidas, com a lista de materiais a partir da camada exterior para a camada interior, em conjunto com as espessuras respectivas. Nesta tabela, $\overline{\#G}$ e $\overline{\#F}$ são o número médio de gerações e o número de avaliações da função objectivo, respectivamente.

4.2.4 Discussão dos Resultados

Este problema de optimização do projecto de uma placa laminada é um problema de natureza combinatória que implica tempos de computação elevados. Em média, cada avaliação

Problema	Materiais					Espessura das Camadas (mm)					$\overline{\#G}$	$\overline{\#F}$
#0	**S**	**N**	**I**	**C**	**B**	0.20	0.20	0.20	0.20	0.20	29	1693
#1	**S**	**N**	**I**	**A**	**G**	0.20	0.16	0.20	0.11	0.13	70	4824
#2	**N**	**S**	**I**	**A**	**G**	0.13	0.20	0.13	0.16	0.17	69	4622
#3	**I**	**S**	**A**	**G**	**R**	0.08	0.07	0.19	0.20	0.20	72	4787
#4	**S**	**N**	**I**	**G**	**R**	0.19	0.08	0.15	0.01	0.07	70	4642
#5	**S**	**N**	**I**	**A**	**R**	0.04	0.05	0.05	0.20	0.20	79	4810
#6	**S**	**I**				0.20	0.19	0.00	0.00	0.00	65	4415

Tabela 4.10: Resultados obtidos para as Instâncias do Problema

da função objectivo demorou cerca de 1.6 segundos. Na Tabela 4.10 é possível verificar o elevado número médio de avaliações da função objectivo necessário para a resolução das diversas instâncias do problema, em especial, quando são consideradas restrições. Para testar o algoritmo, diversas instâncias do problema para as quais se conhecia a solução óptima foram formuladas. Os materiais utilizados nas experiências foram escolhidos para facilitar a avaliação das soluções. As soluções obtidas têm significado físico, validando o modelo matemático.

4.3 Optimização de Placas Laminadas de Materiais Compósitos

O projecto de placas laminadas com materiais compósitos é um problema de optimização de natureza combinatória onde o objectivo é encontrar a sequência óptima de materiais de um dado conjunto, bem como as respectivas orientações das fibras. A optimização do material por natureza e a do dimensionamento por razões tecnológicas, deve ser formulada como um problema de optimização discreta e, assim, resulta num problema de programação inteira.

Uma abordagem baseada em Algoritmos Genéticos (AGs) parece apresentar algumas vantagens na resolução deste problema de optimização estrutural [COIL99, LCOF00, COI$^+$00]. O modelo baseado na maximização da rigidez, optimiza a sequência de lâminas com diferentes orientações das fibras e materiais variáveis; a espessura de cada camada e o número global de camadas são, à priori, fixadas. Além disso, uma restrição no custo global é também considerada. Estuda-se também o comportamento dos AGs em função do número e tipo de elementos finitos usados no cálculo da função objectivo.

Os materiais compósitos têm vindo a receber uma atenção substancial como materiais a utilizar em estruturas de responsabilidade. Apesar da atractividade da elevada relação das propriedades de rigidez face ao peso e da resistência face ao peso dos materiais compósitos, a sua maior vantagem prende-se com a sua capacidade para serem projectados de forma a satisfazer a resistência e a rigidez direccional da estrutura para qualquer tipo de carga simples ou complexa.

No caso das estruturas compósitas laminadas, cada camada apresenta a sua maior rigidez e resistência ao longo da direcção em que as fibras estão orientadas. Pela orientação de cada camada segundo diferentes ângulos, a estrutura pode ser projectada para uma carga específica.

Para além disso, a par com o comportamento estrutural e o peso, o custo é um aspecto de grande interesse, quando se consideram estudos de optimização do projecto de estruturas. Obviamente, reduzindo a quantidade de material necessário para a construção da estrutura, minimiza-se o custo do laminado compósito. No entanto, outra forma de reduzir os custos é a introdução de diversos materiais na sequência de lâminas. Nesse caso é possível utilizar camadas de material mais barato nos locais da estrutura onde o comportamento estrutural é menos importante. As vantagens na utilização de materiais compósitos é de alguma forma neutralizada pela maior complexidade da análise e da optimização estrutural.

Como foi referido, o projecto de estruturas compósitas, envolve geralmente, proble-

mas de programação inteira não convexos e que são, por natureza, discretos. Em geral, o problema de optimização da sequência de lâminas em compósitos laminados tem sido formulado como um problema contínuo e resolvido utilizando técnicas baseadas em gradientes. Estes métodos de resolução apresentam algumas desvantagens:

- o projecto destas estruturas envolve muitas vezes variáveis que estão limitadas a um pequeno conjunto de valores discretos de valores para a espessuras de cada camada, de ângulos de orientação das fibras ou tipos de material, atendendo a limitações de ordem tecnológica ou de custo; nestas condições, estes métodos requerem a transformação destas variáveis em variáveis contínuas, por forma a que uma solução possa ser obtida;

- a conversão das soluções contínuas de novo para os valores discretos possíveis, muitas vezes produz soluções não óptimas, ou mesmo soluções não admissíveis;

- os problemas de laminados compósitos apresentam muitas vezes funções objectivo descontínuas, que exibem múltiplos projectos com comportamento similar, envolvendo muitos óptimos locais.

Apesar dos custos computacionais elevados, os AGs têm vindo a ser aplicados na optimização da sequência de camadas de placas compósitas [CW92], no projecto de painéis compósitos reforçados [NJG+96], ou em laminados [Haj90, LN94, FFJ+98, Haf98].

4.3.1 Análise Estrutural de Laminados

Neste trabalho foi empregue o modelo de camada única equivalente para placas laminadas, baseado na teoria de deformação ao corte de primeira e de terceira ordem, para analisar cada projecto possível. O modelo resultante é adequado para laminados finos e moderadamente espessos, com propriedades materiais que variam arbitrariamente ao longo da espessura, sendo suficientemente eficiente para cálculos em optimização de projecto. Além

disso, esta abordagem permite a redução do número de graus de liberdade requeridos para descrever a resposta estrutural, com uma representação suficientemente detalhada, e sem um custo computacional excessivo.

Nas teorias de camada única equivalente, uma placa laminada heterogénea é tratada como uma lâmina única estaticamente equivalente, com um comportamento constitutivo complexo, o que reduz um problema contínuo tridimensional a um problema bidimensional. Desta forma, os laminados compósitos podem ser modelados com elementos finitos de placa.

O desenvolvimento de uma teoria de placas requer que se adopte uma expansão adequada do campo de deslocamentos da placa. Considerando uma placa compósita laminada de espessura total h, composta de camadas ortotrópicas ou transversalmente isotrópicas, define-se um sistema de coordenadas cartesianas x_i na placa, sendo o plano x_1x_2 coincidente com o plano médio geométrico da placa [LCOF00]. Neste trabalho, foram utilizados na análise placas laminadas compósitas, dois elementos quadriláteros de primeira ordem com oito (família serendipítica) e nove nós (família lagrangeana). Os elementos serendipíticos têm menos nós comparados com os elementos lagrangeanos pois não têm pontos interiores. Todos os elementos apresentam cinco graus de liberdade por nó. O desenvolvimento completo destes elementos é descrito em [Lea98].

4.3.2 Algoritmos Genéticos no Projecto de Placas Laminadas

Na optimização de placas laminadas compósitas, o objectivo é encontrar o material para cada camada e o respectivo ângulo de orientação das fibras que produz a estrutura com melhor comportamento, para condições de carga dadas. Adicionalmente, restrições na geometria, de fabrico, de custo e de falha podem também ser incluídas no projecto.

Para reduzir o espaço de projecto, é considerada uma placa laminada quadrada simétrica e balanceada. Assim, a placa laminada é simétrica em relação ao plano médio, com um ângulo de orientação de $-\alpha^o$ para cada $+\alpha^o$. Para além disso, a espessura global da placa

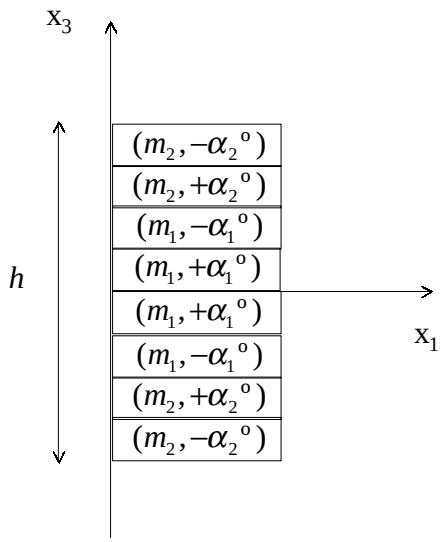

Figura 4.9: Estrutura de uma Placa Laminada Compósita

laminada é fixada inicialmente, variando o material e o ângulo de orientação das fibras em cada camada. A Figura 4.9 ilustra uma placa laminada compósita, onde h é a espessura total da placa, e m_i, α_i são, respectivamente, o material e o ângulo de orientação das fibras na camada i.

Para uma placa com k camadas, os seguintes vectores podem ser definidos: $\alpha = (\alpha_1, \ldots, \alpha_k)$ e $\mathbf{m} = (m_1, \ldots, m_k)$, que são, respectivamente, o vector dos ângulos de orientação das fibras e o vector dos materiais. O problema de optimização estrutural consiste em encontrar a melhor combinação de materiais e ângulos de orientação das fibras que produza a placa mais rígida. A formulação discreta deste modelo, que é obtida pelo método dos elementos finitos, corresponde ao seguinte problema de optimização:

$$\left[\begin{array}{l} \min\limits_{(\mathbf{m},\alpha)\in C} \frac{1}{2}\mathbf{f}^T\mathbf{u} \\ \text{sujeito a} \quad \mathbf{K}(\mathbf{m},\alpha)\mathbf{u} = \mathbf{f} \end{array}\right. \tag{4.3.1}$$

onde $\mathbf{K}(\mathbf{m}, \alpha)$ é a matriz de rigidez, $\mathbf{f}$ é o vector das forças e $\mathbf{u}$ é vector dos deslocamentos. Para além disso, o custo total da placa pode ser restringido, i.e., o custo total deve ser

menor ou igual que um determinado custo P_1, por isso,

$$\sum_{j=1}^{k} m_j c(m_j) \leq P_1 \tag{4.3.2}$$

onde $c(m_j)$ é uma função que devolve o custo relativo do material m_j.

Em conclusão, o problema de optimização da placa pode ser formulado como o seguinte problema de programação inteira,

$$\text{Minimizar} \quad g_1(\mathbf{m}, \alpha) = \frac{1}{2}\mathbf{f}^T\mathbf{u} \tag{4.3.3}$$

$$\text{sujeito a} \quad \mathbf{K}(\mathbf{m}, \alpha)\mathbf{u} = \mathbf{f} \tag{4.3.4}$$

$$g_2(\mathbf{m}, \alpha) = \sum_{j=1}^{k} m_j c(m_j) \leq P_1 \tag{4.3.5}$$

onde $\mathbf{K}(\mathbf{m}, \alpha)$ é a matriz de rigidez, $\mathbf{f}$ é o vector das forças, $\mathbf{u}$ é o vector dos deslocamentos, e g_2 é uma função que corresponde à restrição de custo.

Para ilustrar a aplicação do AG no projecto de placas laminadas compósitas, considere-se o seguinte problema, envolvendo uma placa laminada quadrada simétrica e balanceada apoiada em quatro lados, com dezasseis camadas de $0.15mm$, que podem ser feitas de dois materiais (vidro/*epoxy* (G) e grafite/*epoxy* (C)), e em que as fibras podem ser orientadas segundo sete ângulos (0^o, 15^o, 30^o, 45^o, 60^o, 75^o e 90^o). Uma solução possível para o problema deve especificar, em cada camada, o material e o ângulo de orientação das fibras.

Uma vez que o problema é simétrico e balanceado, com o objectivo de restringir o espaço de busca, apenas quatro variáveis de projecto têm de ser especificadas. Por exemplo, uma solução possível pode ser $[(G, \pm 30^o); (C, \pm 75^o); (G, \pm 45^o); (G, \pm 60^o)]_s$, i.e., a primeira camada é de vidro/*epoxy* com um ângulo de orientação de $+30^o$, a segunda é de vidro/*epoxy* com um ângulo de orientação de -30^o, a terceira camada é de grafite/*epoxy* com um ângulo de orientação de $+75^o$, e assim por diante. A placa laminada é simétrica (subscrito s). É importante notar que, se o ângulo é 0^o ou 90^o, duas camadas idênticas são consideradas.

Representação das Soluções

Considere-se a seguinte placa laminada $[(G, \pm 30^o); (C, \pm 75^o); (G, \pm 45^o); (G, \pm 60^o)]_s$. Esta placa laminada pode ser codificada como

Variável	Material	Ângulo
1	G	30^o
2	C	75^o
3	G	45^o
4	G	60^o

.

Em alternativa pode ser utilizada uma sequência de dígitos, em que cada material é representado por um dígito variando de 0 a 1 (G-0 e C-1), e cada ângulo por outro variando de 0 a 6 (0-0^o, 1-15^o, 2-30^o, 3-45^o, 4-60^o, 5-75^o e 6-90^o), obtendo-se:

$$0\,2 \quad 1\,5 \quad 0\,3 \quad 0\,4 \ .$$

Uma vez que o AG requer a codificação binária das soluções, então,

Variável	Material	Ângulo
1	0	010
2	1	101
3	0	011
4	0	100

.

Finalmente, esta solução seria representada pela seguinte sequência de dígitos binários 0010110100110100.

Cálculo da Função Objectivo

Cada ponto no espaço de procura, i.e., cada cromossoma, é avaliado com base no trabalho feito pelas cargas aplicadas, que representa o inverso da rigidez. O cálculo é efectuado por um módulo de elementos finitos que, para um dado conjunto de materiais e ângulos, produz um valor para a função objectivo. Na abordagem baseada em penalização, as restrições são incluídas na função objectivo através de um termo de penalização levando à seguinte função

$$G(\mathbf{m}, \alpha) = \frac{1}{2}\mathbf{f}^T\mathbf{u} + RC_1(\mathbf{m}, \alpha)^2 \tag{4.3.6}$$

onde R é o coeficiente de penalização e a função $C_1(\mathbf{m}, \alpha)$ é definida por

$$C_1(\mathbf{m}, \alpha) = \min\{0,\ P_1 - g_2(\mathbf{m}, \alpha)\} \tag{4.3.7}$$

Parâmetros

Nesta abordagem, foi utilizada uma população de 100 cromossomas. A população inicial foi gerada aleatoriamente. Como operadores genéticos, considerou-se a recombinação em dois pontos e a mutação uniforme, com probabilidades de 0.7 e 0.001, respectivamente. Em cada geração, os cromossomas foram seleccionados utilizando a selecção uniforme estocástica e considerando um esquema de graduações lineares na avaliação das soluções.

Foram consideradas restrições de custo com os seguintes custos relativos dos materiais: G - 1 e C - 8, i.e., $c(0) = 1$ e $c(1) = 8$. Os cromossomas que representassem soluções não admissíveis foram penalizados. Um esquema de função de penalização foi utilizado para tratar as restrições do problema (este esquema foi descrito anteriormente e é expresso pela equação (3.2.12)). O coeficiente de penalização foi fixado em 1000. Para avaliar a fiabilidade das soluções foram feitas várias execuções. Como critério de paragem, a execução foi terminada quando se observasse a convergência para uma solução ou 100 gerações fossem completadas.

Resultados

Nesta secção são apresentados os resultados obtidos para o problema anteriormente descrito. Além disso, cada instância do problema foi replicada 10 vezes, pelo que, os valores tabelados são valores médios. A Tabela 4.11 apresenta os resultados obtidos (as melhores soluções), em termos do número médio de gerações ($\overline{\#G}$) e de cálculos da função objectivo ($\overline{\#F}$).

A Figura 4.10 apresenta a evolução do valor médio da função objectivo ao longo das gerações, para o caso do problema exemplo sem restrições. Pode ser observado o decréscimo

Custo ($P1$)	$\overline{\#G}$	$\overline{\#F}$	Solução	$f(\mathbf{x})$ (Nm)
-	19	823	$[(C, \pm 45^o)_4]_s$	162
≤ 32	19	823	$[(C, \pm 45^o)_4]_s$	162
≤ 25	21	992	$[(C, \pm 45^o)_2; (G, \pm 45^o)_2;]_s$	173
≤ 18	21	984	$[(C, \pm 45^o)_3; (G, \pm 45^o)]_s$	163
≤ 11	23	1119	$[(C, \pm 45^o); (G, \pm 45^o)_3]_s$	208

Tabela 4.11: Resultados obtidos para o Problema Exemplo

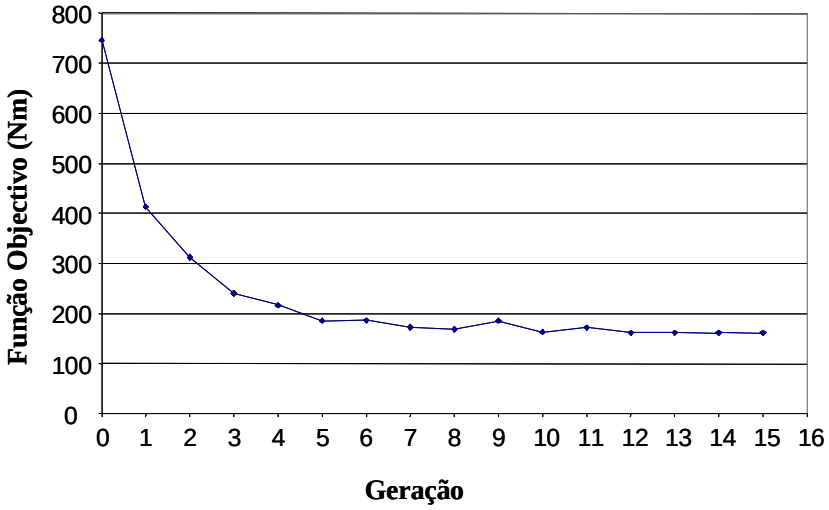

Figura 4.10: Variação do Valor Médio da Função Objectivo ao longo das Gerações

do valor médio da função objectivo dos cromossomas presentes na população ao longo das gerações. Obviamente, esta diminuição ocorre mais acentuadamente no início da procura, verificando-se a tendência para a estabilização do valor médio da função objectivo quando se começa a verificar a convergência da população para uma solução.

4.3.3 Resultados Computacionais

Foram escolhidas diversas aplicações para avaliar:

Problema	Condições de Fronteira	Carregamento
#1	Apoiada nos 4 lados	Carga distribuída
#2	Encastrada nos 4 lados	Carga distribuída
#3	Encastrada em 1 lado	Cargas concentradas nos vértices livres, iguais e opostas

Tabela 4.12: Problemas sem Restrição de Custo

- a fiabilidade dos AGs na optimização de placas laminadas compósitas;
- a influência do número e do tipo de elementos finitos usados no esforço computacional e na precisão dos resultados obtidos;
- a influência da restrição do custo nos resultados obtidos.

Em todos os problemas apresentados, o modelo considera uma placa quadrada de $0.1m$ de lado, formada por dezasseis camadas de $0.15mm$ de espessura, que podem ser feitas de seis tipos de materiais, sendo três de grafite/*epoxy* ($C1$, $C2$ e $C3$) e três de vidro/*epoxy* ($G1$, $G2$ e $G3$). Os ângulos de orientação das fibras estão limitados a um conjunto de sete valores possíveis entre 0^o e 90^o com um incremento de 15^o.

Em todas as tabelas de resultados apresentadas, é indicado o número de elementos, o tipo de elemento - Lagrangeano (L) ou Serendipítico (S) e a teoria de elementos finitos considerada - Mindlin (M) ou Reddy (R).

Problemas sem Restrições

Nesta secção são apresentados os resultados para três problemas de projecto de placas laminadas compósitas. Estes problemas (problemas #1, #2 e #3) diferem nas condições fronteira e no carregamento conforme se apresenta na Tabela 4.12.

O AG foi executado quatro vezes para cada problema sem restrição de custo. Os

Elementos			$\overline{\#G}$	$\overline{\#F}$	Solução
N^o	Tipo	Teo.			
4	L	M	33	1398	$[(C3, \pm 45^o)_4]_s$
16	L	M	27	1339	$[(C3, \pm 45^o)_4]_s$
64	L	M	34	1504	$[(C3, \pm 45^o)_4]_s$
4	S	R	32	1405	$[(C3, \pm 45^o)_4]_s$
16	S	R	30	1295	$[(C3, \pm 45^o)_4]_s$
64	S	R	29	1299	$[(C3, \pm 45^o)_4]_s$
4	L	M	31	1387	$[(C3, \pm 45^o)_3; (G3, \pm 45^o)]_s$
16	L	M	31	1374	$[(C3, \pm 45^o)_4]_s$
64	L	M	34	1419	$[(C3, \pm 45^o)_4]_s$
4	S	R	31	1369	$[(C3, \pm 45^o)_3; (G3, \pm 45^o)]_s$
16	S	R	28	1298	$[(C3, \pm 45^o)_4]_s$

Tabela 4.13: Resultados obtidos para o Problema #1

resultados da aplicação do AG aos problemas são apresentados nas Tabelas 4.13, 4.14 e 4.15. Nestas tabelas são indicados os elementos utilizados, o número médio de gerações, o número médio de cálculos da função objectivo e a melhor solução obtida.

O tipo e número de elementos foi variado com o objectivo de investigar como é que afectam os resultados. Assim, a Figura 4.11 mostra como o tempo de execução varia com o tipo e o número de elementos considerados no cálculo da função objectivo para o Problema #1 . Por outro lado, a Figura 4.12 mostra a variação do valor da função objectivo (o inverso da rigidez) em função do tipo e número de elementos considerados para o Problema #1 . Nestas figuras, apenas foram considerados os resultados obtidos para 4, 16 e 64 elementos serendipíticos e lagrangeanos. O tempo computacional cresce não linearmente com o número de elementos considerado. No entanto, a precisão da solução é, em termos práticos, aproximadamente a mesma para 16 e 64 elementos.

Elementos			$\overline{\#G}$	$\overline{\#F}$	Solução
N^o	Tipo	Teo.			
4	L	M	36	1686	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
16	L	M	31	1370	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
64	L	M	36	1712	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
4	S	R	39	1687	$[(C3, \pm 45^o); (C3, \pm 90^o); (C3, \pm 0^o); (C3, \pm 90^o)]_s$
					$[(C3, \pm 45^o); (C3, \pm 0^o); (C3, \pm 90^o); (C3, \pm 0^o)]_s$
16	S	R	33	1541	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
64	S	R	35	1405	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
4	L	M	38	1681	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
16	L	M	35	1681	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
4	S	R	32	1600	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$
16	S	R	32	1574	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$
					$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$

Tabela 4.14: Resultados obtidos para o Problema #2

Elementos			$\overline{\#G}$	$\overline{\#F}$	Solução
N^o	Tipo	Teo.			
4	L	M	33	1398	$[(C3, \pm 45^o)_4]_s$
16	L	M	27	1339	$[(C3, \pm 45^o)_4]_s$
64	L	M	34	1504	$[(C3, \pm 45^o)_4]_s$
4	S	R	32	1405	$[(C3, \pm 45^o)_4]_s$
16	S	R	30	1295	$[(C3, \pm 45^o)_4]_s$
64	S	R	29	1299	$[(C3, \pm 45^o)_4]_s$
4	L	M	31	1387	$[(C3, \pm 45^o)_4]_s$
16	L	M	31	1374	$[(C3, \pm 45^o)_4]_s$
64	L	M	34	1419	$[(C3, \pm 45^o)_4]_s$
4	S	R	31	1369	$[(C3, \pm 45^o)_4]_s$
16	S	R	28	1298	$[(C3, \pm 45^o)_4]_s$

Tabela 4.15: Resultados obtidos para o Problema #3

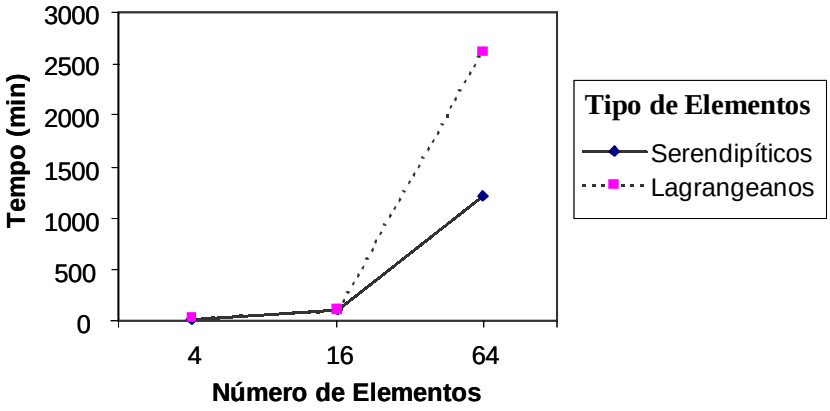

Figura 4.11: Tempo de Execução versus o Número e o Tipo de Elementos Finitos

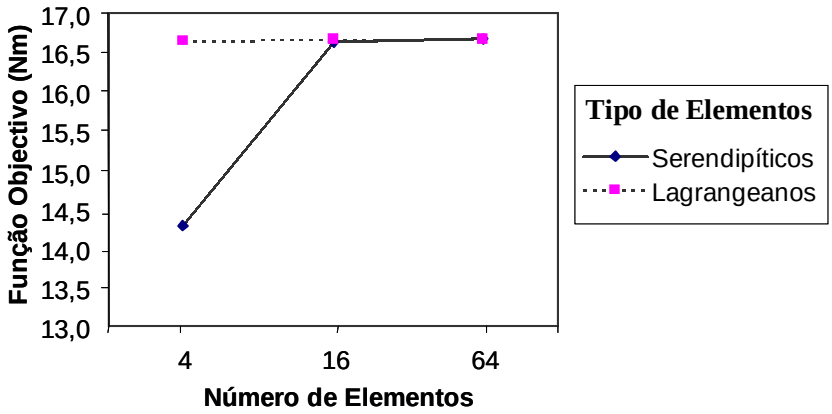

Figura 4.12: Valor da Função Objectivo versus o Número e o Tipo de Elementos Finitos

Problema	Condições de Fronteira	Carregamento
#4	Apoiado nos 4 lados	Carga distribuída
#5	Encastrado nos 4 lados	Carga distribuída

Tabela 4.16: Problemas com Restrição de Custo

Problemas com Restrições

Nesta secção são apresentados os resultados para dois problemas de projecto de placas laminadas compósitas. Estes problemas (problemas #4 e #5) diferem nas condições fronteira e no carregamento conforme se apresenta na Tabela 4.16.

Nestes problemas são consideradas restrições de custo, sendo os custos relativos dos seis materiais indicados na Tabela 4.17. Os custos relativos apresentados correspondem a $c(0) = 1$, $c(1) = 3$, $c(2) = 4$, $c(3) = 8$, $c(4) = 10$ e $c(5) = 12$. Os mesmos parâmetros do AG foram considerados com excepção do tamanho da população que foi neste caso de 150 cromossomas.

Os resultados da aplicação do AG aos problemas são apresentados nas Tabelas 4.18 e 4.19. Nestas tabelas são indicados os elementos utilizados, o número médio de gerações, o número médio de cálculos da função objectivo e a melhor solução obtida.

Material	Fibra	Custo Relativo ($c(m_j)$)
$G1$	Vidro/*Epoxy*	1
$G2$	Vidro/*Epoxy*	3
$G3$	Vidro/*Epoxy*	4
$C1$	Grafite/*Epoxy*	8
$C2$	Grafite/*Epoxy*	10
$C3$	Grafite/*Epoxy*	12

Tabela 4.17: Lista de Materiais Compósitos

Custo	Elementos			$\overline{\#G}$	$\overline{\#F}$	Solução	$f(\mathbf{x})$
($P1$)	N^o	Tipo	Teo.				(Nm)
-	4	L	M	31	2090	$[(C3, \pm 45^o)_4]_s$	16.6
	4	S	M	29	1245	$[(C3, \pm 45^o)_3; (G3, \pm 45^o)]_s$	14.3
≤ 40	4	L	M	39	2244	$[(C3, \pm 45^o)_3; (G3, \pm 45^o)]_s$	16.8
	4	S	M	37	2156	$[(C3, \pm 45^o)_3; (G3, \pm 45^o)]_s$	14.3
≤ 30	4	L	M	37	2860	$[(C2, \pm 45^o)_2; (C1, \pm 45^o); (G1, \pm 45^o)]_s$	18.4
	4	S	M	42	2614	$[(C3, \pm 45^o)_2; (G2, \pm 45^o)_2]_s$	15.1
≤ 20	4	L	M	37	2866	$[(C2, \pm 45^o); (C1, \pm 45^o); (G1, \pm 45^o)_2]_s$	20.2
	4	S	M	41	2879	$[(C2, \pm 45^o); (C1, \pm 45^o); (G1, \pm 45^o)_2]_s$	17.3
≤ 10	4	L	M	31	2262	$[(G2, \pm 45^o)_3; (G1, \pm 45^o)]_s$	42.0
	4	S	M	32	2249	$[(G2, \pm 45^o)_3; (G1, \pm 45^o)]_s$	30.1

Tabela 4.18: Resultados obtidos para o Problema #4

Custo	Elementos			$\overline{\#G}$	$\overline{\#F}$	Solução	$f(\mathbf{x})$
($P1$)	N^o	Tipo	Teo.				(Nm)
-	4	L	M	40	2605	$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$	4.7
	4	S	M	34	2495	$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (C3, \pm 90^o)]_s$	4.7
						$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (C3, \pm 0^o)]_s$	
≤ 40	4	L	M	44	2916	$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (G3, \pm 0^o)]_s$	4.8
	4	S	M	37	2452	$[(C3, \pm 0^o); (C3, \pm 90^o)_2; (G3, \pm 0^o)]_s$	4.8
						$[(C3, \pm 90^o); (C3, \pm 0^o)_2; (G3, \pm 90^o)]_s$	
≤ 30	4	L	M	45	2037	$[(C3, \pm 90^o); (C3, \pm 0^o); (G2, \pm 0^o); (G2, \pm 90^o)]_s$	5.1
	4	S	M	43	3058	$[(C3, \pm 90^o); (C3, \pm 0^o); (G2, \pm 0^o); (G2, \pm 90^o)]_s$	5.1
						$[(C3, \pm 0^o); (C3, \pm 90^o); (G2, \pm 90^o); (G2, \pm 0^o)]_s$	
≤ 20	4	L	M	40	2928	$[(C2, \pm 90^o); (C1, \pm 0^o); (G1, \pm 90^o)_2]_s$	5.8
						$[(C2, \pm 0^o); (C1, \pm 90^o); (G1, \pm 0^o)_2]_s$	
	4	S	M	55	3486	$[(C2, \pm 90^o); (C1, \pm 0^o); (G1, \pm 90^o)_2]_s$	5.8
≤ 10	4	L	M	35	2428	$[(G2, \pm 90^o); (G2, \pm 0^o)_2; (G1, \pm 90^o)]_s$	10.8
						$[(G2, \pm 0^o); (G2, \pm 90^o)_2; (G1, \pm 0^o)]_s$	
	4	S	M	28	2203	$[(G2, \pm 90^o); (G2, \pm 0^o)_2; (G1, \pm 90^o)]_s$	10.8
						$[(G2, \pm 0^o); (G2, \pm 90^o)_2; (G1, \pm 0^o)]_s$	

Tabela 4.19: Resultados obtidos para o Problema #5

4.3.4 Discussão dos Resultados

Os resultados apresentados mostram que os AGs são uma técnica válida para o projecto de placas laminadas compósitas, produzindo resultados com significado físico.

A análise dos resultados para o problema sem restrição de custo permite concluir que:

- para os problemas #1 e #3, o AG encontra a solução óptima na maior parte dos casos;
- para o problema #2, existe uma grande dispersão de resultados como consequência de, por um lado, o problema ter vários óptimos globais e, por outro, o modelo de elementos finitos ser menos satisfatório.

De notar a influência do número de graus de liberdade do problema no tempo global necessário para encontrar a solução. De notar também, como era de esperar, que os elementos finitos de primeira ordem conduzem a menores tempos de execução do que os elementos baseados na teoria de Reddy.

Em termos gerais, o custo computacional cresce não linearmente com o número de elementos considerados, i.e., com a precisão imposta no código de elementos finitos. No entanto, os resultados indicam que uma solução obtida com 16 elementos é suficientemente precisa, com tempos computacionais razoáveis.

Parece contraditório que o número de cálculos da função objectivo não aumente de forma monótona à medida que a restrição de custo é minimizada. No entanto, deve notar-se que este é um problema combinatório, e o conjunto de pontos admissíveis para diferentes instâncias do custo, pode ser bastante diferente. Logo, o número de cálculos da função objectivo (a média de 10 execuções) reflecte esta diferença.

Parte II

Optimização Multi-objectivo

Capítulo 5

Introdução

Esta segunda parte relativa a Optimização Multi-objectivo (OM) é iniciada com um capítulo introdutório que pretende apresentar a área da optimização multi-objectivo. Assim, na Secção 5.1 são apresentadas quer a formulação matemática de problemas multi-objectivo quer diversos conceitos associados à optimização multi-objectivo. Nesta mesma secção descrevem-se também as condições de optimalidade para o caso multi-objectivo, bem como alguns dos algoritmos tradicionais para a optimização multi-objectivo. As principais abordagens evolucionárias desenvolvidas para optimização multi-objectivo são descritas na Secção 5.2. Algumas considerações são feitas no que diz respeito à geração de problemas teste e comparação de algoritmos no âmbito da optimização multi-objectivo.

Em seguida, nos Capítulos 6 e 7 são apresentadas duas abordagens evolucionárias para optimização multi-objectivo, um Algoritmo Genético Elitista e uma Estratégia Evolutiva Elitista.

Finalmente, o Capítulo 8 encerra esta segunda parte com a apresentação de algumas aplicações de AEs a problemas de optimização multi-objectivo. Na Secção 8.1 é descrita a aplicação de um AG ao projecto de placas laminadas com materiais isotrópicos. Na última secção (Secção 8.2) é feita a descrição da aplicação de um AG ao projecto de placas

laminadas com materiais compósitos.

5.1 Optimização Multi-objectivo

Os problemas de optimização multi-objectivo são comuns no mundo real, embora, tradicionalmente, estes sejam formulados como problemas uni-objectivo. Por exemplo, na compra de um automóvel diversos objectivos podem ser considerados. Por um lado, é desejável adquirir um automóvel que apresente um bom desempenho (em termos de, por exemplo, da sua velocidade de ponta e da sua capacidade de aceleração); por outro lado, o automóvel deve ser seguro, económico e apresentar o menor custo de aquisição possível. A resolução de problemas multi-objectivo deste tipo é uma tarefa difícil uma vez que, em geral, para esta classe de problemas, os objectivos são conflituosos num espaço multi-dimensional. Nestes problemas não existe apenas uma única solução, dado que, a interacção entre os múltiplos objectivos faz com que exista um conjunto de soluções eficientes (estas são conhecidas como soluções óptimas de Pareto como será indicado mais adiante).

5.1.1 Formulação Matemática

Matematicamente, um problema de optimização multi-objectivo é a minimização ou maximização de um conjunto de funções sujeitas a restrições nas variáveis [SNT85, Zit99]. A seguinte notação irá ser considerada:

- $\mathbf{x}$ é o vector das *variáveis de decisão*;
- $\mathbf{f}(\mathbf{x})$ é o vector das *funções objectivo*, onde cada componente é uma função que se pretende maximizar ou minimizar;
- $\mathbf{g}(\mathbf{x})$ é o vector das *restrições do tipo desigualdade* que devem ser satisfeitas;
- $\mathbf{h}(\mathbf{x})$ é o vector das *restrições do tipo igualdade* que devem ser satisfeitas.

Utilizando esta notação, um problema de minimização pode ser formulado genericamente da seguinte forma:

$$\min \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_s(\mathbf{x})) \text{ onde } \mathbf{x} \in \Omega \tag{5.1.1}$$

sujeito a

$$g_j(\mathbf{x}) \geq 0 \text{ com } j = 1, \ldots, m$$

$$h_i(\mathbf{x}) = 0 \text{ com } i = m+1, \ldots, m+p$$

Na formulação apresentada existem s funções objectivo, n variáveis reais, m restrições do tipo desigualdade e p restrições do tipo igualdade. Tal como na optimização uni-objectivo, as restrições do tipo desigualdade e igualdade definem a região admissível.

Definição 5.1.1. (**Região Admissível**) *A região admissível é o conjunto de vectores de variáveis de decisão que satisfazem as restrições do problema, i.e.,*

$$A = \{\mathbf{x} \in \Omega : \mathbf{g}(\mathbf{x}) \geq 0 \wedge \mathbf{h}(\mathbf{x}) = 0\}.$$

A noção de optimalidade em problemas multi-objectivo deve ter presente que uma solução que minimiza um objectivo poderá não minimizar todos os outros. Por isso, uma solução óptima com respeito a um objectivo é, em geral, não óptima para com os outros objectivos. Nesta situação, existe um conjunto de soluções óptimas que traduzem compromissos entre os objectivos do problema. As soluções óptimas do problema multi-objectivo são necessariamente pontos admissíveis, i.e., pertencem à região admissível, satisfazendo todas as restrições.

O *espaço das variáveis* $\Omega \subseteq \Re^n$ corresponde ao conjunto de todos os valores possíveis para as variáveis de decisão. Por outro lado, a imagem de todos os valores possíveis para as variáveis de decisão que satisfazem as restrições, o conjunto A, constitui o *espaço dos objectivos*. Um ponto admissível no *espaço das variáveis* ($\mathbf{x} \in A$) representa uma solução

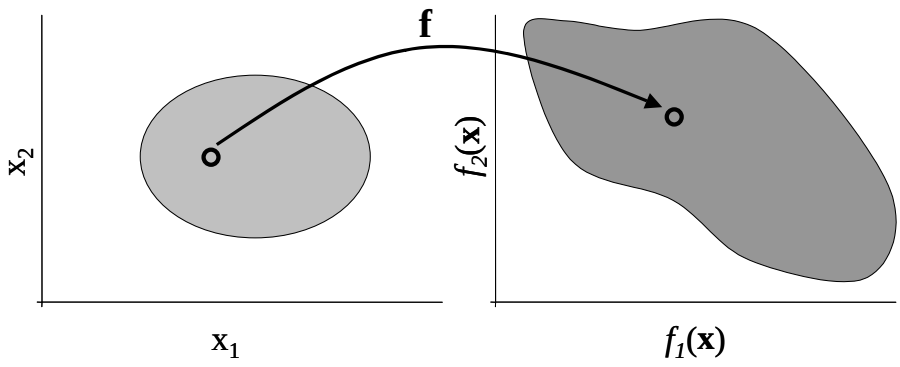

Figura 5.1: Espaço das Variáveis e Espaço dos Objectivos

a que corresponde um ponto no *espaço dos objectivos*, cuja qualidade é determinada em termos dos valores das funções objectivo. Esta situação é ilustrada na Figura 5.1, para o caso de $n = 2$, $s = 2$, $m = 0$ e $p = 0$. A função $\mathbf{f}$ permite determinar o ponto no espaço dos objectivos $\mathbf{f}(\mathbf{x})$ que é imagem do vector de variáveis de decisão $\mathbf{x}$ do espaço das variáveis de decisão.

Sem perda de generalidade, a formulação (5.1.1) refere-se a problemas de optimização multi-objectivo em que todos os objectivos se pretendem minimizar. No entanto, em optimização multi-objectivo, três tipos de situações podem ocorrer:

- minimização de todos os objectivos;
- maximização de todos os objectivos;
- minimização de alguns e maximização de outros objectivos.

Por razões de simplicidade, em geral, todas as funções são convertidas ou para a forma de maximização ou para a forma de minimização. Por exemplo, para converter um problema multi-objectivo em que se pretende maximizar todos ou parte dos objectivos num problema de minimização como o formulado em (5.1.1) basta reformular todos os objectivos de maximização da seguinte forma $\max f(\mathbf{x}) = -\min(-f(\mathbf{x}))$. Nesta dissertação, sem perda

de generalidade e por simplificação, considerar-se-á sempre o caso de minimização de todos os objectivos.

Num problema multi-objectivo com s objectivos, se os óptimos de cada um dos objectivos considerados individualmente forem diferentes, então diz-se que os objectivos são *conflituosos*. É esta situação que torna os problemas multi-objectivo particularmente difíceis, não existindo uma solução óptima única para o problema mas sim um conjunto de soluções óptimas. Caso contrário, quando os objectivos não são conflituosos, as soluções óptimas de cada um dos objectivos considerados individualmente coincidem e, o problema pode ser resolvido por uma técnica de optimização uni-objectivo. Como ilustração do caso em que existem diversos objectivos conflituosos, considere-se o problema da compra de um automóvel, em que os objectivos a minimizar são: o tempo de aceleração e o custo de aquisição. Para este problema, os dois objectivos são conflituosos, uma vez que as soluções óptimas de cada um dos objectivos considerados individualmente não coincidem, i.e., os automóveis com maior capacidade de aceleração (menor tempo de aceleração) têm, em geral, custos mais elevados.

5.1.2 Conceitos Básicos

Em optimização uni-objectivo, o espaço de procura é completamente (totalmente) ordenado em termos da função objectivo uma vez que, para quaisquer duas soluções $\mathbf{x}$ e $\mathbf{y}$, se verifica $f(\mathbf{x}) \leq f(\mathbf{y})$ ou $f(\mathbf{y}) < f(\mathbf{x})$. No entanto, quando mais do que um objectivo é considerado, o espaço de procura não é completamente ordenado, mas sim parcialmente ordenado. Este facto pode ser ilustrado considerando-se os pontos representados no espaço dos objectivos correspondentes às soluções $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ e $\mathbf{d}$ como é mostrado na Figura 5.2. A solução representada pelo ponto $\mathbf{a}$ é melhor nos dois objectivos do que a representada pelo ponto $\mathbf{c}$. Por outro lado, a solução $\mathbf{b}$ é preferível à $\mathbf{d}$, uma vez que é melhor no segundo objectivo. Na generalidade dos problemas multi-objectivo não existe uma ordenação com-

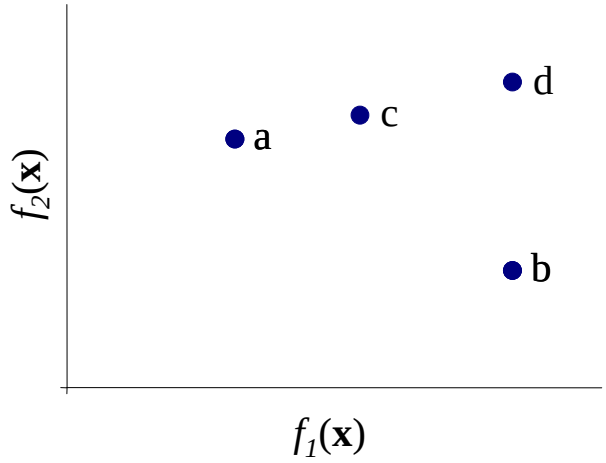

Figura 5.2: Representação de Soluções no Espaço dos Objectivos

pleta na base das funções objectivo e, por isso, a noção de optimalidade é diferente da de problemas uni-objectivo. Para expressar as situações ilustradas na Figura 5.2, as relações $=$, $\leq$ e $<$ podem ser extendidas a vectores como é descrito em seguida.

Definição 5.1.2. *Para quaisquer dois vectores* $\mathbf{x}$ *e* $\mathbf{y}$*,*

$$\mathbf{x} = \mathbf{y} \text{ se } \forall_{i\in\{1,2,\ldots,s\}} : x_i = y_i$$

$$\mathbf{x} \leq \mathbf{y} \text{ se } \forall_{i\in\{1,2,\ldots,s\}} : x_i \leq y_i$$

$$\mathbf{x} < \mathbf{y} \text{ se } \mathbf{x} \leq \mathbf{y} \wedge \mathbf{x} \neq \mathbf{y}.$$

As relações $\geq$ *e* $>$ *podem ser definidas de forma semelhante.*

Utilizando a anterior definição, pode-se afirmar que $\mathbf{a} < \mathbf{c}$, $\mathbf{c} < \mathbf{d}$ e, consequentemente, $\mathbf{a} < \mathbf{d}$. No entanto, quando se comparam as soluções $\mathbf{a}$ e $\mathbf{b}$ não se pode afirmar que uma é superior à outra ou vice-versa, uma vez que $\mathbf{a} \not< \mathbf{b}$ e $\mathbf{b} \not< \mathbf{a}$. Por um lado, a solução $\mathbf{a}$ é melhor no primeiro objectivo mas a solução $\mathbf{b}$ é melhor no segundo. Assim, quaisquer dois vectores de variáveis de decisão, $\mathbf{x}$ e $\mathbf{y}$, relacionam-se de uma das três seguintes formas: $\mathbf{f}(\mathbf{x}) \leq \mathbf{f}(\mathbf{y})$, $\mathbf{f}(\mathbf{y}) \leq \mathbf{f}(\mathbf{x})$, ou $\mathbf{f}(\mathbf{x}) \not\leq \mathbf{f}(\mathbf{y}) \wedge \mathbf{f}(\mathbf{y}) \not\leq \mathbf{f}(\mathbf{x})$.

Com base na relação atrás descrita pode-se introduzir a seguinte definição de dominância de Pareto.

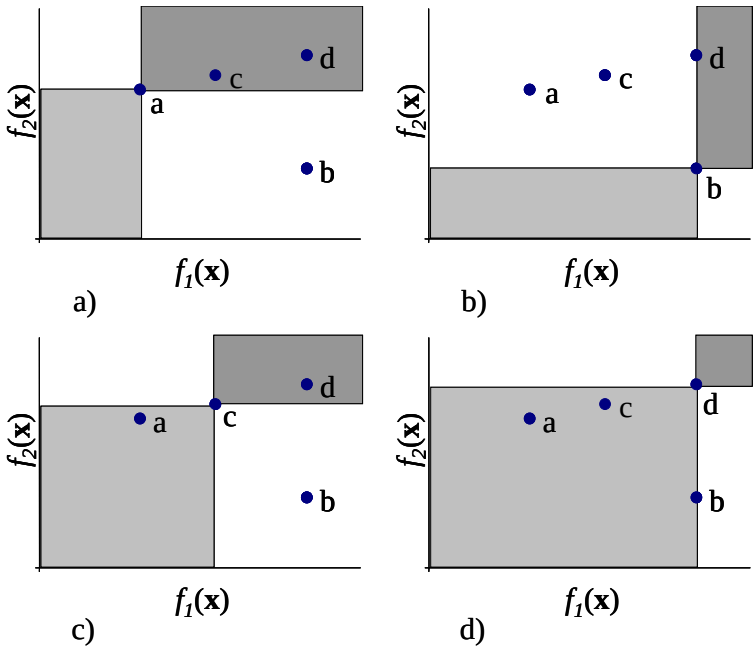

Figura 5.3: Relação de Dominância entre Soluções

Definição 5.1.3. (**Dominância de Pareto**) *Para quaisquer dois vectores de variáveis de decisão* $\mathbf{x}$ *e* $\mathbf{y}$*,*

$$\mathbf{x} \prec \mathbf{y} \text{ } se \text{ } \mathbf{f}(\mathbf{x}) < \mathbf{f}(\mathbf{y}) \text{ } (\mathbf{x} \text{ } domina \text{ } \mathbf{y})$$

$$\mathbf{x} \sim \mathbf{y} \text{ } se \text{ } \mathbf{f}(\mathbf{x}) \not\leq \mathbf{f}(\mathbf{y}) \wedge \mathbf{f}(\mathbf{y}) \not\leq \mathbf{f}(\mathbf{x}) \text{ } (\mathbf{x} \text{ } é \text{ } indiferente \text{ } a \text{ } \mathbf{y})$$

A definição de dominância de Pareto para problemas de maximização é idêntica.

Para além do conceito de dominância atrás definido, o conceito de dominância fraca é também bastante utilizado. Para dois vectores de variáveis de decisão $\mathbf{x}$ e $\mathbf{y}$, diz-se que $\mathbf{x}$ *domina fracamente* $\mathbf{y}$ ($\mathbf{x} \preceq \mathbf{y}$) se $\forall_{i \in \{1,2,\ldots,s\}} : f_i(\mathbf{x}) < f_i(\mathbf{y})$.

As Figuras 5.3a), 5.3b), 5.3c) e 5.3d) apresentam, assinaladas a cinzento escuro, respectivamente, as regiões do espaço dos objectivos dominadas por cada uma das soluções **a**, **b**, **c** e **d**. Para além disso, para cada solução, são assinaladas a cinzento claro as regiões do espaço dos objectivos que contêm soluções que as dominam. Para cada solução, todas

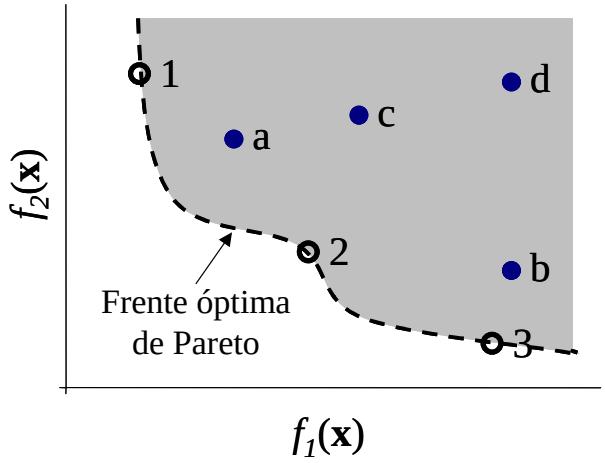

Figura 5.4: Optimalidade de Pareto no Espaço dos Objectivos

as soluções que não pertençam às regiões assinaladas a cinzento são soluções indiferentes. A partir da figura, pode-se verificar que:

- a solução **a** domina as soluções **c** e **d** (e é indiferente a **b**);
- a solução **b** domina a solução **d** (e é indiferente a **a** e **c**);
- a solução **c** domina a solução **d** e é dominada pela solução **a** (e é indiferente a **b**);
- a solução **d** é dominada pelas soluções **a**, **b** e **c**.

Deste conjunto de quatro soluções, apenas as soluções **a** e **b** não são dominadas por nenhuma das outras. Refira-se que, em termos de dominância fraca, a solução **d** não é dominada pela solução **b**, uma vez que a solução **b** não é estritamente melhor, em todos os objectivos, que a solução **d**.

Com base no conceito de dominância de Pareto, pode-se definir o critério de optimalidade para problemas multi-objectivo. Uma solução **x** é uma solução óptima de Pareto se não for dominada por nenhuma outra solução admissível. As soluções óptimas de Pareto situam-se numa região onde não é possível melhorar qualquer dos objectivos sem degradar

pelo menos um dos outros objectivos. Por isso, qualquer solução nesta região não é dominada por qualquer outra. Estas soluções óptimas de Pareto também são muitas vezes chamadas de soluções eficientes ou soluções não inferiores.

Definição 5.1.4. (**Optimalidade de Pareto**) *Um vector de variáveis de decisão* $\mathbf{x}$ *é não dominado em relação a um conjunto* $B \subseteq A$ *se,*

$$\neg\exists_{\mathbf{y}\in B} : \mathbf{y} \prec \mathbf{x}$$

Um vector de variáveis de decisão $\mathbf{x}$ *é uma solução óptima de Pareto se e só se* $\mathbf{x}$ *é não dominada no conjunto* A*, o conjunto de soluções admissíveis.*

Na Figura 5.4 as soluções **1**, **2** e **3** são soluções óptimas de Pareto. Estas soluções são indiferentes umas em relação às outras. Ao contrário dos problemas com um único objectivo, não existe uma solução óptima única, mas sim um conjunto de soluções eficientes que traduzem diferentes compromissos entre os objectivos. O conjunto das imagens no espaço dos objectivos de todas as soluções óptimas de Pareto constitui a superfície ou frente óptima de Pareto (representada na Figura 5.4 pela linha descontínua). Em termos de *dominância fraca*, um vector de variáveis de decisão $\mathbf{x}$ é não dominado em relação a um conjunto $B \subseteq A$ se $\neg\exists_{\mathbf{y}\in B}\forall_{i\in\{1,2,\ldots,s\}} : f_i(\mathbf{y}) < f_i(\mathbf{x})$.

Definição 5.1.5. (**Conjuntos de Soluções Não Dominadas e Frentes**) *A função* $p(B)$ *devolve o conjunto de vectores de variáveis de decisão não dominados em* B*:*

$$p(B) = \{\mathbf{x} \in B : \mathbf{x} \text{ é não dominado em } B\}$$

O conjunto $p(B)$ *é o conjunto de soluções não dominadas em* B *e o conjunto correspondente de vectores de objectivos* $\mathbf{f}(p(B))$ *é a frente de Pareto em relação a* B*. Para o conjunto* A *das soluções admissíveis,* $p(A)$ *constitui o conjunto de soluções óptimas de Pareto e o conjunto correspondente de vectores de objectivos* $\mathbf{f}(p(A))$ *é a frente óptima de Pareto.*

Tal como nos problemas com um único objectivo em que poderão existir óptimos locais, nos problemas multi-objectivo também podem existir conjuntos de soluções óptimas locais que constituem um conjunto de soluções não dominadas numa determinada vizinhança.

A convexidade duma frente de Pareto pode ser definida da seguinte forma.

Definição 5.1.6. (**Convexidade da Frente de Pareto**) *Uma frente de Pareto* $\mathbf{f}(p(B))$ *é convexa se, para quaisquer dois vectores de objectivos,* $\mathbf{a}$ *e* $\mathbf{b}$*, pertencentes a* $\mathbf{f}(p(B))$*, existe um vector de objectivos* $\mathbf{c}$ *tal que* $\mathbf{c} \leq \alpha\mathbf{a} + (1-\alpha)\mathbf{b}$*, para todo* $\alpha \in [0,1]$*. Caso contrário,* $\mathbf{f}(p(B))$ *é côncava.*

Muitas vezes, o conjunto de soluções não dominadas define frentes de Pareto que são convexas em determinadas regiões do espaço dos objectivos, e não convexas (côncavas) noutras regiões (tal como sucede com a frente de Pareto apresentada na Figura 5.4).

5.1.3 Condições de Optimalidade

Nesta secção serão apresentadas as condições de optimalidade para problemas de optimização como os formulados em (5.1.1) [SNT85]. Contudo, apenas são apresentadas as condições de optimalidade para problemas multi-objectivo com restrições do tipo desigualdade. As demonstrações dos teoremas que serão apresentados nesta secção podem ser encontradas em [SNT85].

Quer as condições necessárias de optimalidade, quer as condições suficientes de optimalidade são apresentadas assumindo-se que todas as funções objectivo $f_i(\mathbf{x})$, para $i = 1, \ldots, s$, e, $g_j(\mathbf{x})$, para $j = 1, \ldots, m$, são continuamente diferenciáveis.

Assumindo-se que $\mathbf{x}^*$ é uma solução óptima de Pareto, podem-se deduzir as condições necessárias de optimalidade.

Teorema 5.1.1. *(**Condições Necessárias para a Optimalidade de Pareto**) Se* $\mathbf{x}^*$

é uma solução óptima de Pareto, então existe $\mu \geq 0$ *e* $\lambda \geq 0$ *tal que*

$$\sum_{i=1}^{s} \mu_i \nabla f_i(\mathbf{x}^*) - \sum_{j=1}^{m} \lambda_j \nabla g_j(\mathbf{x}^*) = 0 \ e \ \sum_{j=1}^{m} \lambda_j g_j(\mathbf{x}^*) = 0.$$

Para problemas multi-objectivo sem restrições, o teorema anterior requer apenas a seguinte condição:

$$\sum_{i=1}^{s} \mu_i \nabla f_i(\mathbf{x}^*) = 0. \tag{5.1.2}$$

Se um ponto $\mathbf{x}^*$ é uma solução óptima de Pareto, então existe $\mu > 0$ que verifica a condição (5.1.2). Para um problema com s objectivos e n variáveis de decisão, a anterior condição corresponde, na forma matricial, à seguinte equação:

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_s}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_s}{\partial x_2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_n} & \frac{\partial f_2}{\partial x_n} & \cdots & \frac{\partial f_s}{\partial x_n} \end{pmatrix} \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_s \end{pmatrix} = 0.$$

Em geral, as condições necessárias são utilizadas para mostrar que um determinado ponto não é uma solução óptima de Pareto. Por outro lado, as condições suficientes que serão enunciadas em seguida, se se verificarem num ponto $\mathbf{x}^*$, garantem que $\mathbf{x}^*$ é uma solução óptima de Pareto.

Teorema 5.1.2. *(**Condições Suficientes para a Optimalidade de Pareto**) Se todas as funções objectivo* $f_i(\mathbf{x})$*, para* $i = 1, \ldots, s$*, são convexas e todas as funções das restrições* $g_j(\mathbf{x})$*, para* $j = 1, \ldots, m$*, são não convexas, então uma solução admissível* $\mathbf{x}^*$ *que verifique*

$$\sum_{i=1}^{s} \mu_i \nabla f_i(\mathbf{x}^*) - \sum_{j=1}^{m} \lambda_j \nabla g_j(\mathbf{x}^*) = 0 \ e \ \sum_{j=1}^{m} \lambda_j g_j(\mathbf{x}^*) = 0 \ para \ \mu > 0 \ e \ \lambda \geq 0,$$

é uma solução óptima de Pareto.

5.1.4 Procura e Tomada de Decisão

Ao resolver um problema multi-objectivo, duas fases distintas podem ser consideradas: a procura e a tomada de decisão. A procura refere-se ao processo de optimização que permite

encontrar um conjunto de soluções que aproxima o conjunto de soluções óptimas de Pareto para o problema considerado. A tomada de decisão corresponde à selecção de uma solução que é um compromisso entre os objectivos do problema a partir do conjunto de soluções que aproxima o conjunto de soluções óptimas de Pareto. Um decisor humano é necessário para executar a tarefa, muitas vezes difícil, de escolha de soluções compromisso.

Dependendo da forma como as duas fases anteriormente referidas são combinadas, os métodos para optimização multi-objectivo podem ser classificados em três categorias distintas [YCL95]:

Tomada de decisão antes da procura - neste caso, os objectivos de um problema multi-objectivo são agregados num único objectivo que inclui informação de preferência fornecida pelo decisor antes de se fazer a procura;

Procura antes da tomada de decisão - nesta situação, a procura é feita sem nenhuma informação de preferência dada pelo decisor. O decisor escolhe a partir do resultado da procura que corresponde a um conjunto de soluções de compromisso;

Tomada de decisão durante a procura - neste último caso, o decisor pode articular as suas preferências durante o processo de procura de uma forma interactiva. O processo de procura é feito por passos, no final dos quais, um conjunto de soluções de compromisso é apresentada ao decisor que fornece informações de preferência que por sua vez permitem orientar a procura.

Os algoritmos para optimização multi-objectivo considerados nesta dissertação pertencem à segunda categoria atrás referida, i.e., numa primeira fase, a procura de soluções é efectuada, cabendo, em seguida, ao decisor a escolha das soluções pretendidas. Logo, os algoritmos desta categoria devem ser capazes de, por um lado, procurar soluções em espaços de procura complexos e, por outro, gerar um conjunto de boas aproximações às soluções óptimas de Pareto.

5.1.5 Algoritmos Tradicionais

Os algoritmos tradicionais para optimização multi-objectivo, em geral, agregam os diferentes objectivos num único objectivo e depois o problema é resolvido utilizando uma técnica de optimização uni-objectivo. A agregação dos diversos objectivos num único objectivo implica que este seja parametrizado por forma a que se reflicta a importância dos diversos objectivos. Neste aspecto, a formulação de um único objectivo agregando os múltiplos objectivos conflituosos pode ser considerada análoga à classe de métodos englobados pela primeira categoria referida na secção anterior (Tomada de decisão antes da procura), em que o decisor fornece informação relacionada com as suas preferências antes da procura. No entanto, em geral, os parâmetros da função objectivo agregada são variados sistematicamente em vez de fixados pelo decisor de acordo com as suas preferências, i.e., diversas execuções são efectuadas com diferentes combinações dos valores dos parâmetros com o objectivo de obter uma boa aproximação ao conjunto de soluções óptimas de Pareto. Uma desvantagem óbvia destas abordagens é a necessidade de efectuar múltiplas execuções para encontrar diferentes soluções aproximando o conjunto de soluções óptimas de Pareto.

Alguns dos algoritmos tradicionais mais conhecidos são o método dos pesos e o método das restrições [Coh78]. Descrições de outras abordagens tradicionais para optimização multi-objectivo, nomeadamente, métodos baseados em métricas pesadas e métodos baseados em programação por metas, podem ser encontradas em [Deb01, AC01].

Método dos Pesos

Neste método, o problema multi-objectivo original é convertido num problema uni-objectivo pela agregação dos diversos objectivos numa única função objectivo. A função objectivo agregada é uma combinação linear dos diversos objectivos [Coh78]:

$$\min f(\mathbf{x}) = w_1 f_1(\mathbf{x}) + \ldots + w_s f_s(\mathbf{x}).$$

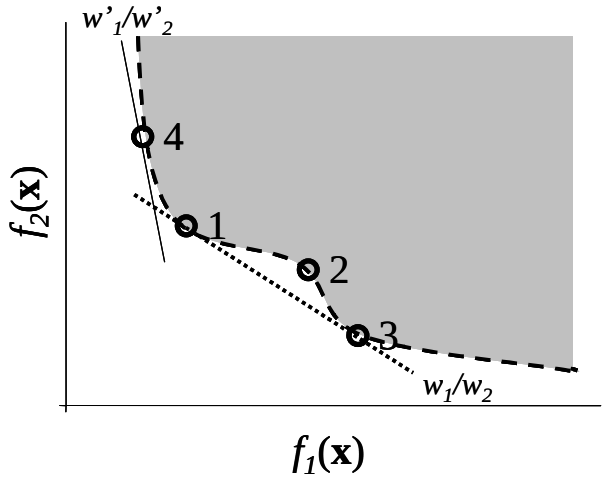

Figura 5.5: Método dos Pesos

Os coeficientes w_i, com $i = 1, \ldots, s$, correspondem aos pesos normalizados de tal forma que $\sum w_i = 1$. A resolução do problema uni-objectivo, com um certo número de diferentes combinações de pesos, permite obter um conjunto de soluções. No entanto, a principal desvantagem deste método é que não consegue encontrar todas as soluções óptimas de Pareto para problemas que apresentem frentes não convexas. Para além disso, a obtenção de múltiplas soluções requer diversas execuções do algoritmo de optimização uni-objectivo considerado. Por outro lado, certos problemas são sensíveis à escolha dos pesos, podendo não ser fácil obter uma boa distribuição das soluções no espaço dos objectivos. O método dos pesos é ilustrado na Figura 5.5. Ao fixar pesos w_1 e w_2 para os dois objectivos, o processo de optimização consiste em deslocar a recta com declive $-w_1/w_2$ ao longo do espaço dos objectivos até que nenhuma solução esteja abaixo dela e pelo menos uma seja interceptada por ela. A recta a tracejado, correspondendo à combinação de pesos w_1 e w_2 para os dois objectivos, permite obter as soluções 1 e 3. No entanto, a solução 2 nunca será encontrada por este método qualquer que seja a combinação de pesos, uma vez que pertence a uma porção da frente que é não convexa. A combinação de pesos w'_1 e w'_2 define

a recta com declive $-w_1'/w_2'$ que permite obter a solução 4 pertencente a uma porção convexa da frente.

Método das Restrições

Esta técnica transforma $s-1$ objectivos do problema multi-objectivo original (com s objectivos) em restrições. Os objectivos a transformar em restrições podem ser escolhidos arbitrariamente resultando no seguinte problema uni-objectivo [Coh78]:

$$\begin{aligned} &\min f(\mathbf{x}) = f_j(\mathbf{x}) \\ &\text{sujeito a } f_i(\mathbf{x}) \leq \varepsilon_i \text{ com } 1 \leq i \leq s, i \neq j \end{aligned}.$$

Os limites superiores das restrições, ε_i, são coeficientes que são variados por forma a encontrar diferentes soluções óptimas de Pareto. Ao contrário do método anterior, este consegue encontrar soluções associadas a partes não convexas da frente de Pareto. No entanto, existe o problema da escolha apropriada dos valores de ε_i, que poderá não ser simples. Determinados valores de ε_i poderão tornar a região admissível do problema uni-objectivo vazia, i.e., não existirá nenhuma solução para o correspondente problema uni-objectivo. Para evitar esta situação, ter-se-ão que escolher intervalos de valores adequados para estes coeficientes o que, dependendo do problema, poderá ser complexo. A Figura 5.6 ilustra a aplicação do método das restrições a um problema com dois objectivos, considerando-se a minimização do primeiro objectivo $f_1(\mathbf{x})$ sujeito a uma restrição no segundo objectivo $f_2(\mathbf{x}) \leq \varepsilon_2$. Como é mostrado na Figura 5.6, a solução 2, pertencente à porção não convexa da frente, pode ser determinada impondo uma restrição no segundo objectivo com limite superior ε_2. Variando o limite superior da restrição, é possível obter diferentes soluções, em sucessivas execuções do algoritmo de optimização uni-objectivo. No entanto, situações como a ilustrada na figura, quando foi fixado como limite superior da restrição ε_2', podem ocorrer. Nesta situação, a região admissível do problema uni-objectivo é vazia, não sendo possível obter nenhuma solução. Portanto, muitas vezes, a escolha de limites superiores

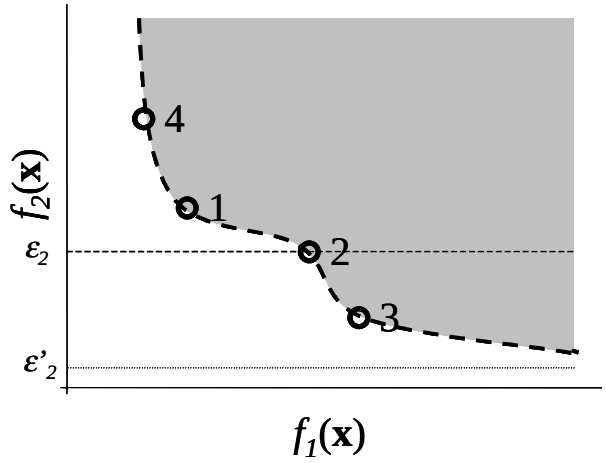

Figura 5.6: Método dos Restrições

da restrição ou das restrições adequados não é fácil, implicando o conhecimento prévio de informação relativa ao problema que se está a resolver. Esta dificuldade torna-se particularmente evidente quando existem mais de dois objectivos, aumentando o número de possíveis combinações dos limites das restrições.

5.2 Abordagens Evolucionárias

As abordagens tradicionais, como o método dos pesos e o método das restrições referidos na Secção 5.1.5, apresentam algumas dificuldades:

- algumas são sensíveis à forma da frente de soluções óptimas de Pareto do problema;
- a informação requerida relativa ao problema não é conhecida.

Outros potenciais problemas são mencionadas por Deb [Deb01], como, por exemplo, o facto de que as abordagens tradicionais requerem mais do que uma execução para obter uma aproximação a uma única solução óptima de Pareto e de que a qualidade da aproximação obtida depende das características do algoritmo de optimização uni-objectivo utilizado.

Uma diferença crucial entre os algoritmos tradicionais e as abordagens evolucionárias é que estas trabalham com populações de soluções em cada geração. Esta diferença constitui um vantagem enorme uma vez que, em optimização multi-objectivo, um dos objectivos é encontrar o maior número possível de soluções óptimas de Pareto (ou, pelo menos, um número suficiente de soluções que permitam ao decisor escolher uma solução compromisso). A utilização de uma população de soluções permite que, numa única execução, sejam encontradas aproximações a múltiplas soluções óptimas de Pareto. Para além disso, elimina-se a necessidade de reformular o problema multi-objectivo como um problema uni-objectivo, recorrendo a artifícios como a introdução de pesos, restrições, ou outros parâmetros. Por outro lado, um outro objectivo da optimização multi-objectivo é encontrar um conjunto bem distribuído de aproximações às soluções óptimas de Pareto. Uma vez que as abordagens evolucionárias podem utilizar mecanismos para preservar a diversidade das soluções na população, é possível encontrar um conjunto de soluções bem distribuídas no espaço dos objectivos. De salientar que, ao contrário das abordagens tradicionais, os AEs não necessitam de diversas execuções para a obtenção de um conjunto de soluções aproximadas às soluções óptimas de Pareto.

Assim, com o objectivo de encontrar um conjunto de soluções bem distribuídas no espaço dos objectivos que seja uma boa aproximação ao conjunto de soluções óptimas de Pareto, diversas abordagens evolucionárias foram desenvolvidas, encontrando-se descritas na literatura [Deb01]. Face à importância do elitismo na procura e, uma vez que a generalidade de AEs mais recentes o incluem de alguma forma, em geral, duas classes de algoritmos são distinguidas: os que não utilizam explicitamente elitismo e os que o utilizam. Os primeiros AEs surgidos não utilizavam explicitamente elitismo, nomeadamente, o *Vector Evaluated Genetic Algorithm* (VEGA) de Schaffer [Sch85], o *Multiple Objective Genetic Algorithm* (MOGA) de Fonseca e Fleming [FF93], o *Niched Pareto Genetic Algorithm* (NPGA) de Horn *et al.* [HNG94] e o *Nondominated Sorting Genetic Algorithm*

(NSGA) de Srinivas e Deb [SD94].

O VEGA foi uma das primeiras abordagens evolucionárias para optimização multi-objectivo. A sua implementação relativamente simples constitui uma extensão de um AG para optimização uni-objectivo ao âmbito da optimização multi-objectivo. Assim, para tratar os diversos objectivos de um problema multi-objectivo, a população é dividida, de forma aleatória, em sub-populações em número igual ao número de objectivos do problema. A cada sub-população é associado um único objectivo. Posteriormente, as soluções de cada uma das sub-populações são avaliadas de acordo com os correspondentes objectivos. A selecção é aplicada de forma independente a cada sub-população. Novas soluções são geradas utilizando os operadores usuais de recombinação e mutação. Ao contrário das implementações mais recentes, o VEGA não utiliza nenhum mecanismo de controlo da diversidade das soluções na população para além da usual mutação.

As abordagens evolucionárias mais recentes, o MOGA, o NPGA e o NSGA basearam-se, em alguns aspectos, nas sugestões feitas por Goldberg [Gol89]. Goldberg sugeriu que se utilizasse o conceito de dominância para guiar a procura. Assim, as soluções não dominadas numa população deveriam tender a gerar mais descendentes. Para além disso, para tratar a questão da manutenção da diversidade das soluções na população, um esquema baseado em *sharing* foi proposto. Os três algoritmos referidos (o MOGA, o NPGA e o NSGA) seguem estas sugestões, embora difiram na forma como o desempenho das soluções é medido. No MOGA, um esquema baseado em graduações é utilizado para medir o desempenho dos indivíduos. A graduação de um dado indivíduo da população é proporcional ao número de indivíduos da população que são dominados por ele. Um esquema de *sharing* com as distâncias medidas no espaço dos objectivos é utilizado para distribuir as soluções ao longo da frente de Pareto. No NPGA, por outro lado, um esquema de selecção por torneio baseado no conceito de dominância de Pareto foi implementado. Assim, a selecção consiste em comparar dois indivíduos escolhidos aleatoriamente da população com um sub-conjunto

de indivíduos da população (também, escolhido aleatoriamente e, tipicamente, correspondendo a 10% da população). Se um dos dois indivíduos é dominado pelos indivíduos da sub-população e o outro não é, então o indivíduo não dominado é seleccionado. Por outro lado, quando ambos indivíduos são dominados ou não dominados, o indivíduo a seleccionar é escolhido através de um esquema de *sharing* no espaço dos objectivos. No NSGA, os indivíduos da população são classificados com base na dominância, definindo diversos grupos de indivíduos que constituem diferentes frentes. Para cada uma das frentes, é aplicado um mecanismo de *sharing* com as distâncias medidas no espaço das variáveis. Uma função auxiliar é utilizada para medir o desempenho dos indivíduos da população.

Apesar do sucesso da aplicação destas abordagens a diversos problemas de optimização multi-objectivo, alguns autores [ZDT00, VVL00] sugeriram que o elitismo pode melhorar o desempenho dos AEs e, também, impedir que soluções eficientes encontradas durante a procura sejam perdidas. Tal como no caso dos AEs para optimização uni-objectivo, diferentes níveis de elitismo podem ser utilizados. No entanto, como em optimização multi-objectivo existem conjuntos de soluções não dominadas de igual importância para a procura e o número de soluções desses conjuntos é muito variável, a introdução de elitismo levanta algumas questões [Zit99], tais como:

- Quais os indivíduos a arquivar e durante quanto tempo?
- Quando e como devem ser os indivíduos arquivados re-inseridos na população?

No que diz respeito ao elitismo, há duas formas principais de o implementar. A primeira consiste na utilização de um arquivo (uma população secundária) onde são mantidas soluções não dominadas encontradas ao longo da procura. Nesta implementação é necessário definir a forma de actualização, tamanho e estrutura da população secundária, bem como a forma como as soluções são re-inseridas na população. Outra possibilidade, é a utilização de esquemas de selecção que implementem elitismo como, por exemplo, a

selecção $(\mu + \lambda)$ das EEs. A abordagem elitista, baseada em população secundária, poderá ser vantajosa em problemas em que o conjunto de soluções óptimas de Pareto tenha uma cardinalidade alta. Para além disso, é possível parametrizar o elitismo, variando, por exemplo, o esquema de actualização da população secundária e o nível de elitismo. No entanto, a actualização da população secundária requer um esforço computacional adicional.

Uma das primeiras abordagens elitistas em optimização multi-objectivo foi o *Strength Pareto Evolutionary Algorithm* (SPEA) de Zitzler e Thiele [ZT99]. Nesta abordagem é mantido um arquivo, contendo as soluções não dominadas encontradas anteriormente. Em cada geração, os indivíduos não dominados são copiados para o arquivo. Para cada indivíduo do arquivo é calculada uma graduação proporcional ao número de indivíduos que são dominados por ele. A medida do desempenho dum dado indivíduo da população é calculada de acordo com as graduações dos indivíduos do arquivo que o dominam. Para além disso, uma técnica de *clustering* é utilizada para manter diversidade. Posteriormente, Deb *et al.* [DAPM00] propuseram o *Elitist Non-dominated Sorting Genetic Algorithm* (NSGA-II) mais eficiente, em termos computacionais, do que o original NSGA. Em cada geração, após a definição de frentes de forma similar ao NSGA, o NSGA-II utiliza uma selecção determinística que implementa elitismo. Para além disso, um esquema baseado em *crowding* garante diversidade das soluções na população. Duas versões do NSGA-II foram implementadas, uma utilizando a tradicional representação binária das variáveis de decisão, e outra considerando uma representação real das variáveis de decisão. Estas duas abordagens elitistas são baseadas em AGs. Por outro lado, implementações baseadas em EEs são mais raras, tal como o algoritmo desenvolvido por Kursawe [Kur90] e o *Pareto Archived Evolution Strategy* (PAES) proposto por Knowles e Corne [KC00]. O algoritmo desenvolvido por Kursawe consiste numa EE modificada para resolver problemas multi-objectivo. Neste algoritmo, cada objectivo tem uma probabilidade associada (estas probabilidades podem variar ao longo da procura). A selecção de um indivíduo é feita de acordo com um

dos objectivos que, por sua vez, é escolhido com base nas probabilidades associadas aos objectivos. Cada indivíduo é representado por dois vectores de variáveis de decisão, um dominante e um recessivo. A medida do desempenho é calculada com base numa combinação pesada dos valores da função objectivo do vector dominante e do recessivo. Como operador genético é utilizada a usual mutação e um operador que troca os elementos dos vectores dominantes com os recessivos (de forma análoga ao operador de recombinação discreta). O PAES, por sua vez, consiste numa EE-(1+1) combinada com um arquivo que guarda algumas das soluções não dominadas encontradas ao longo da procura. Cada novo indivíduo, resultante da aplicação da mutação ao progenitor, é comparado com as soluções presentes no arquivo. O PAES utiliza um mecanismo de *crowding*, baseado na definição de uma grelha no espaço dos objectivos. A diversidade é conseguida com base na contagem do número de soluções em cada uma das células da grelha. Neste algoritmo, as variáveis de decisão, ao contrário das tradicionais EEs, são representadas utilizando sequências de dígitos binários. Por outro lado, não é considerada a adaptação, ao longo da procura, dos parâmetros de mutação. Para além disso, as versões (1+λ) e (μ + λ), propostas pelos mesmos autores, não apresentaram, em geral, melhor desempenho do que a versão (1+1) do algoritmo.

5.3 Problemas Teste

Diversos autores têm proposto diferentes problemas para testar o desempenho dos algoritmos na resolução de problemas multi-objectivo. Os problemas multi-objectivo apresentados na Tabela 5.1 foram escolhidos a partir de um número significativo de trabalhos anteriores na área [Sch85, FF98, Pol97, Kur90, ZDT00]. Estes problemas irão ser considerados para testar os algoritmos propostos. Todos os problemas têm duas funções objectivo a minimizar e nenhuma restrição. A Tabela 5.1 descreve estes problemas, mostrando o número

de variáveis (n), os seus limites, as soluções óptimas de Pareto, e a natureza da frente de soluções óptimas de Pareto para cada problema.

O projecto de problemas teste para optimização multi-objectivo tem sido alvo de investigação por parte de alguns investigadores. Esta preocupação em projectar problemas multi-objectivo resulta do facto de se pretender que os problemas teste possuam características bem definidas e sejam conhecidas as suas soluções (conjuntos de soluções óptimas de Pareto). Este não é o caso de muitos dos problemas reais em que, em geral, não são conhecidas, à partida, as suas características e as suas soluções. Alguns autores propuseram procedimentos para a geração sistemática de problemas teste com características bem definidas. Este é o caso do procedimento proposto por Deb [Deb99] para a geração de problemas teste possuindo frentes com diferentes características de convexidade, de continuidade, de uniformidade da distribuição das soluções no espaço e na frente de soluções óptimas de Pareto. São exemplos disso, os problemas ZDT1 a ZDT6 apresentados na Tabela 5.1. Para além disso, diversos problemas teste com restrições são também propostos [Deb99]. No entanto, uma vez que os algoritmos propostos apenas irão ser testados para problemas sem restrições, aqui não são apresentados quaisquer problemas multi-objectivo com restrições. Mais recentemente, também utilizando procedimentos de geração sistemática, Deb *et al.* [DTLZ02] propuseram problemas com mais de dois objectivos. A Tabela 5.2 apresenta alguns desses problemas sem restrições e três objectivos.

5.4 Medidas de Comparação dos Algoritmos

A comparação de diferentes algoritmos de optimização multi-objectivo é substancialmente mais complexa do que no caso de algoritmos de optimização uni-objectivo, porque o problema em si mesmo consiste em encontrar um conjunto de soluções não dominadas que seja:

Problema	Funções Objectivo	Sol. Óptimas de Pareto
SCH ($n = 1$) $x \in [-10^3, 10^3]$	$f_1(x) = x^2$ $f_2(x) = (x-2)^2$	$x \in [0, 2]$ (frente convexa)
FON ($n = 3$) $x_1, \ldots, x_n \in [-4, 4]$	$f_1(\mathbf{x}) = 1 - \exp(-\sum_{i=1}^{3}(x_i - \frac{1}{\sqrt{3}})^2)$ $f_2(\mathbf{x}) = 1 - \exp(-\sum_{i=1}^{3}(x_i + \frac{1}{\sqrt{3}})^2)$	$x_1 = x_2 = x_3 \in [-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}]$ (frente não convexa)
POL ($n = 2$) $x_1, \ldots, x_n \in [-\pi, \pi]$	$f_1(\mathbf{x}) = [1 + (A_1 - B_1)^2 + (A_2 - B_2)^2]$ $f_2(\mathbf{x}) = [(x_1 + 3)^2 + (x_2 + 1)^2]$ $A_1 = 0.5 \sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$ $A_2 = 1.5 \sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$ $B_1 = 0.5 \sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$ $B_2 = 1.5 \sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$	(frente não convexa e descontínua)
KUR ($n = 3$) $x_1, \ldots, x_n \in [-5, 5]$	$f_1(\mathbf{x}) = \sum_{i=1}^{n-1}(-10\exp(-0.2\sqrt{x_i^2 + x_{i+1}^2}))$ $f_2(\mathbf{x}) = \sum_{i=1}^{n}(\lvert x_i \rvert^{0.8} + 5\sin x_i^3)$	(frente não convexa)
ZDT1 ($n = 30$) $x_1, \ldots, x_n \in [0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})[1 - \sqrt{x_1/g(\mathbf{x})}]$ $g(\mathbf{x}) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	$x_1 \in [0, 1]$ $x_2 = \ldots = x_n = 0$ (frente convexa)
ZDT2 ($n = 30$) $x_1, \ldots, x_n \in [0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})[1 - (x_1/g(\mathbf{x}))^2]$ $g(\mathbf{x}) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	$x_1 \in [0, 1]$ $x_2 = \ldots = x_n = 0$ (frente não convexa)
ZDT3 ($n = 30$) $x_1, \ldots, x_n \in [0, 1]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})[1 - \sqrt{x_1/g(\mathbf{x})} - \frac{x_i}{g(\mathbf{x})}\sin(10\pi x_1)]$ $g(\mathbf{x}) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	$x_1 \in [0, 1]$ $x_2 = \ldots = x_n = 0$ (frente convexa e descontínua)
ZDT4 ($n = 10$) $x_1 \in [0, 1]$ $x_2, \ldots, x_n \in [-5, 5]$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})[1 - \sqrt{x_1/g(\mathbf{x})}]$ $g(\mathbf{x}) = 1 + 10(n-1) + \sum_{i=2}^{n}[x_i^2 - 10\cos(4\pi x_i)]$	$x_1 \in [0, 1]$ $x_2 = \ldots = x_n = 0$ (frente não convexa)
ZDT5 ($n = 11$) $x_1 \in \{0, 1\}^{30}$ $x_2, \ldots, x_n \in \{0, 1\}^5$	$f_1(\mathbf{x}) = 1 + u(x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x})(1/x_1)$ $g(\mathbf{x}) = \sum_{i=2}^{n}[v(u(x_i))]$ $v(u(x_i)) = \begin{cases} 2 + u(x_i) & \text{se } u(x_i) < 5 \\ 1 & \text{se } u(x_i) \geq 5 \end{cases}$ $u(x_i)$ devolve o número de uns de x_i	$x_1 \in \{0, 1\}^{30}$ $u(x_2) = \ldots = u(x_n) = 0$ (frente convexa)
ZDT6 ($n = 10$) $x_1, \ldots, x_n \in [0, 1]$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1)\sin^6(4\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x})[1 - (f_1(x)/g(\mathbf{x}))^2]$ $g(\mathbf{x}) = 1 + 9[(\sum_{i=2}^{n} x_i)/(n-1)]^{0.25}$	$x_1 \in [0, 1]$ $x_2 = \ldots = x_n = 0$ (frente não convexa, e espaçamento não uniforme)

Tabela 5.1: Problemas Multi-objectivo (2 objectivos)

Problema	Funções Objectivo
DTLZ1 ($n = 7$)	$f_1(\mathbf{x}) = \frac{1}{2}x_1x_2(1 + g(\mathbf{x}))$
$x_1, \ldots, x_n \in [0, 1]$	$f_2(\mathbf{x}) = \frac{1}{2}x_1(1 - x_2)(1 + g(\mathbf{x}))$
	$f_3(\mathbf{x}x) = \frac{1}{2}(1 - x_1)(1 + g(\mathbf{x}))$
	$g(\mathbf{x}) = 100\left(5 + \sum_{i=3}^{7}((x_i - 0.5)^2 - cos(20\pi(x_i - 0.5)))\right)$
DTLZ2 ($n = 12$)	$f_1(\mathbf{x}) = cos(x_1\pi/2)cos(x_2\pi/2)(1 + g(\mathbf{x}x))$
$x_1, \ldots, x_n \in [0, 1]$	$f_2(\mathbf{x}) = cos(x_1\pi/2)sin(x_2\pi/2)(1 + g(\mathbf{x}))$
	$f_3(\mathbf{x}) = sin(x_1\pi/2)(1 + g(\mathbf{x}))$
	$g(\mathbf{x}) = \sum_{i=3}^{12}(x_i - 0.5)^2$
DTLZ3 ($n = 12$)	$f_1(\mathbf{x}) = cos(x_1\pi/2)cos(x_2\pi/2)(1 + g(\mathbf{x}))$
$x_1, \ldots, x_n \in [0, 1]$	$f_2(\mathbf{x}) = cos(x_1\pi/2)sin(x_2\pi/2)(1 + g(\mathbf{x}))$
	$f_3(\mathbf{x}) = sin(x_1\pi/2)(1 + g(\mathbf{x}))$
	$g(\mathbf{x}) = 100\left(10 + \sum_{i=3}^{12}((x_i - 0.5)^2 - cos(20\pi(x_i - 0.5)))\right)$
DTLZ4 ($n = 12$)	$f_1(\mathbf{x}) = cos(x_1^{100}\pi/2)cos(x_2^{100}\pi/2)(1 + g(\mathbf{x}))$
$x_1, \ldots, x_n \in [0, 1]$	$f_2(\mathbf{x}) = cos(x_1^{100}\pi/2)sin(x_2^{100}\pi/2)(1 + g(\mathbf{x}))$
	$f_3(\mathbf{x}) = sin(x_1^{100}\pi/2)(1 + g(\mathbf{x}))$
	$g(\mathbf{x}) = \sum_{i=3}^{12}(x_i^{100} - 0.5)^2$

Tabela 5.2: Problemas Multi-objectivo (3 objectivos)

- uma boa aproximação ao verdadeiro conjunto de soluções óptimas de Pareto (a distância entre o conjunto aproximado e o verdadeiro deve ser minimizada);
- bem distribuído no espaço dos objectivos.

Diversas tentativas podem ser encontradas na literatura para expressar as afirmações anteriores por meio de métricas quantitativas [Deb01]. A métrica que é considerada nesta dissertação para comparar o desempenho dos algoritmos é a descrita por Knowles e Corne [KC00] e é baseada num método estatístico proposto por Fonseca e Fleming [FF95]. A ideia básica é comparar as aproximações à frente óptima de Pareto obtidas por diferentes algoritmos através da definição de superfícies. A definição de cada uma das superfícies resulta da ligação das soluções encontradas por cada um dos algoritmos através de rectas. Desta forma, cada uma das superfícies divide o espaço dos objectivos em duas regiões: uma contendo as soluções obtidas pelo algoritmo, e outra contendo todos os pontos que

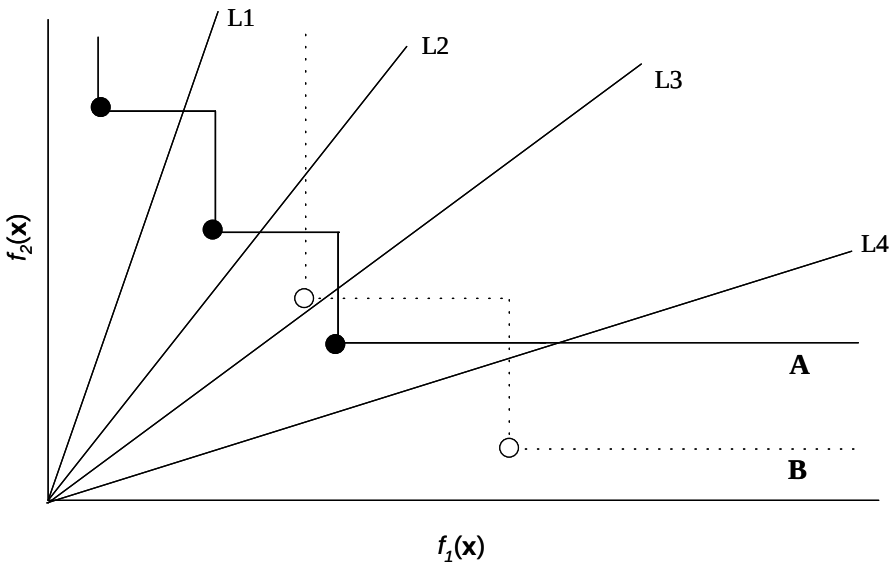

Figura 5.7: Amostragem da Frente de Pareto

são dominados. Uma colecção de rectas de amostragem que intersectam as superfícies é considerada para comparar, em diferentes regiões do espaço dos objectivos, o desempenho dos algoritmos. Para cada linha de amostragem, um algoritmo ultrapassa o outro se a intersecção da linha de amostragem com a respectiva superfície está mais próxima da origem. A Figura 5.7 ilustra a definição de superfícies para as aproximações à frente de Pareto obtidas por dois algoritmos A e B (correspondendo às superfícies A e B, respectivamente). As linhas L1 a L4 são utilizadas para comparar o desempenho dos algoritmos em diferentes regiões do espaço dos objectivos. Por exemplo, as linhas L1 e L2 intersectam a superfície A em pontos mais próximos da origem do que a superfície B, indicando a superioridade do algoritmo A face ao algoritmo B nesta região do espaço dos objectivos. O contrário ocorre para as linhas L3 e L4, onde o algoritmo B ultrapassa o algoritmo A. Então, para várias execuções dos algoritmos, um teste estatístico baseada no teste *Rank-sum* de *Mann-Whitney* é aplicado para os dados previamente recolhidos. Os resultados de uma comparação podem ser apresentados num par de valores $[a, b]$, onde a é a percentagem

do espaço dos objectivos para o qual o algoritmo A foi estatisticamente superior a B, e b é percentagem similar para o algoritmo B. Logo, a é a percentagem do espaço dos objectivos onde o algoritmo A não é "batido" e, b é a percentagem do espaço dos objectivos onde o algoritmo B não é "batido". Por consequência, tipicamente, se $a \approx b \approx 100\%$ então os algoritmos A e B têm resultados semelhantes.

Capítulo 6

Um Algoritmo Genético Elitista

Na década passada, os Algoritmos Genéticos (AGs) [Gol89] foram extendidos com o objectivo de tratar problemas multi-objectivo, como os trabalhos de Schaffer [Sch85], Fonseca e Fleming [FF95], Horn *et al.* [HNG94], Srinivas e Deb [SD94] e Zitzler e Thiele [ZT99]. Tal como já foi referido, estas abordagens multi-objectivo exploram algumas das características dos AGs para resolver estes problemas, em particular:

- uma vez que os AGs trabalham com populações de potenciais soluções podem, em princípio, encontrar, numa única execução, diversas soluções que aproximem as soluções óptimas de Pareto;
- os AGs, utilizando mecanismos que permitem controlar a diversidade, podem encontrar uma gama alargada de soluções que aproximem a fronteira óptima de Pareto.

Apesar do sucesso da aplicação destas abordagens a diversos problemas de optimização multi-objectivo, alguns autores [ZDT00, VVL00] sugeriram que o elitismo pode melhorar o desempenho dos AGs e prevenir também que soluções eficientes encontradas durante a procura sejam perdidas. Por este motivo, um novo esquema elitista foi desenvolvido e implementado, constituindo o *Multiobjective Elitist Genetic Algorithm* (MEGA) [CO02a].

6.1 Medição do Desempenho

Um dos primeiros AEs, proposto para optimização multi-objectivo, foi o *Non-dominated Sorting Genetic Algorithm* (NSGA) de Srinivas e Deb [SD94]. A principal diferença do NSGA em relação aos AGs convencionais diz respeito à medição do desempenho das soluções que permite guiar a procura em direcção ao conjunto de soluções óptimas de Pareto, valorizando as soluções não dominadas. Em cada geração, a não dominância é testada, definindo uma aproximação ao conjunto de soluções óptimas de Pareto. Por outro lado, um mecanismo de *sharing* é utilizado para distribuir as soluções da população ao longo da região no espaço dos objectivos onde se encontram as soluções óptimas de Pareto.

O MEGA é, em termos de medição do desempenho das soluções, semelhante ao NSGA, mas utilizando uma técnica elitista. Logo, em cada geração, todas as soluções não dominadas constituem a primeira frente. A estas soluções é atribuído um valor da medida do desempenho igual ao tamanho da população P. Para manter diversidade na população, um esquema de *sharing* é então aplicado aos valores da medida do desempenho dessas soluções [DG89]. Portanto, a medida do desempenho de cada solução é dividida por uma quantidade proporcional ao número de soluções que estejam a uma distância inferior a σ_{share}, i.e., ao número de soluções que pertençam a esse nicho. O parâmetro σ_{share} é a distância máxima entre soluções necessária para formar uma frente bem distribuída. Em seguida, as soluções da primeira frente são ignoradas temporariamente, e o resto das soluções são processadas. Ao segundo nível de soluções não dominadas é atribuído um valor da medida do desempenho igual ao menor valor da medida do desempenho das soluções da primeira frente menos um. Para cada solução da segunda frente, o valor da medida do desempenho é então dividido por uma quantidade proporcional ao número de soluções que pertençam ao seu nicho. Este processo é repetido até que todas as soluções tenham atribuído um valor da medida do desempenho. Este processo de medição do desempenho enfatiza as

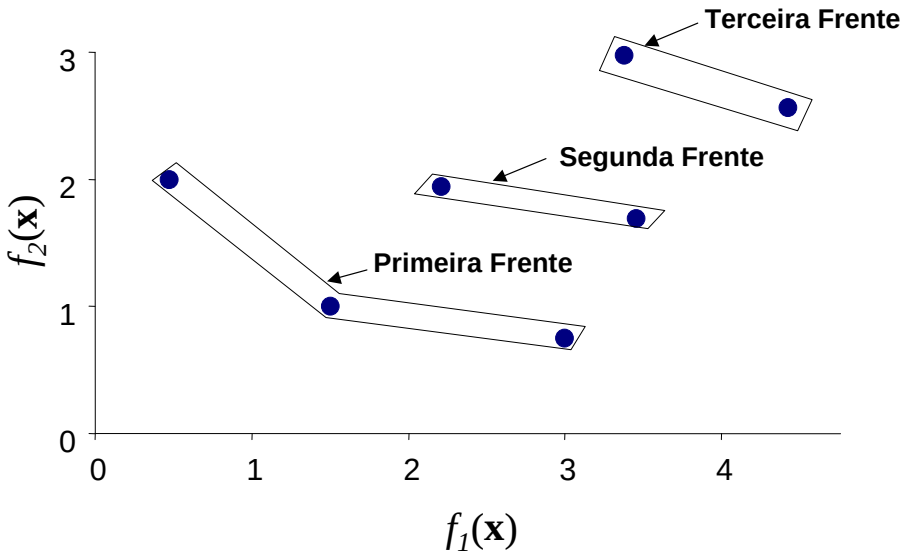

Figura 6.1: Distribuição das Soluções por Frentes

soluções não dominadas, uma vez que todas as soluções da primeira frente têm um valor de medida do desempenho superior ao de todas as soluções da segunda frente, e assim por diante. Para além disso, a coexistência de múltiplas soluções não dominadas é encorajada pelo esquema de *sharing* (Figura 6.1).

O *sharing* pode ser implementado utilizando a distância genotípica (medida no espaço das variáveis) ou a distância fenotípica (medida no espaço dos objectivos). No que diz respeito à medição das distâncias entre soluções, existem vários argumentos que levam a preferir a primeira abordagem (distâncias ao nível do espaço das variáveis) relativamente à outra (distâncias ao nível do espaço dos objectivos). A possibilidade da existência de múltiplas soluções no espaço das variáveis com o mesmo valor da função objectivo favorece a primeira abordagem quando a diversidade no espaço das variáveis é de primeira importância; no entanto, o conjunto de soluções óptimas de Pareto é definido no espaço dos objectivos e, portanto, pode ser de maior importância ter diversidade ao nível do espaço dos objectivos, i.e., uma boa distribuição das soluções no espaço dos objectivos.

Nesta implementação adoptou-se o *sharing* com medição das distâncias entre soluções ao nível do espaço dos objectivos. A selecção e os operadores genéticos de recombinação e mutação permanecem idênticos aos utilizados nos AGs convencionais para optimização uni-objectivo.

6.2 Elitismo com População Secundária

Apesar do sucesso da aplicação de AEs multi-objectivo, alguns autores, como Zitzler *et al.* [ZDT00], sugerem que o elitismo pode aumentar o seu desempenho, para além de prevenir a perda de boas soluções encontradas ao longo da procura. A técnica elitista desenvolvida é baseada numa População Secundária (PS) que contém todas as potenciais soluções óptimas de Pareto encontradas até ao momento durante a procura. Neste sentido, a PS é completamente independente da população principal, e, no final da procura, contém o conjunto de todas as soluções não dominadas encontradas durante a procura.

Um parâmetro θ é introduzido para controlar o nível de elitismo. Este parâmetro representa o número máximo de soluções não dominadas na PS, a chamada elite, que é introduzida na população principal. Estes pontos não dominados participam efectivamente no processo de procura. Se o número total de soluções na PS (n_{PS}) é maior que θ, então θ soluções não dominadas são escolhidas aleatoriamente da PS para formar a elite. Caso contrário, apenas n_{SP} soluções não dominadas são seleccionadas da PS para constituir a elite. Neste último caso, a elite terá apenas n_{PS} membros.

Na sua forma mais simples, em cada geração, as novas potenciais soluções óptimas de Pareto são guardadas na PS ao longo das gerações. A actualização da PS implica a determinação da optimalidade de Pareto para todas as soluções guardadas até ao momento, por forma a eliminar aquelas que se tornem dominadas. À medida que o tamanho da PS aumenta, o tempo necessário para completar esta operação pode tornar-se significativo.

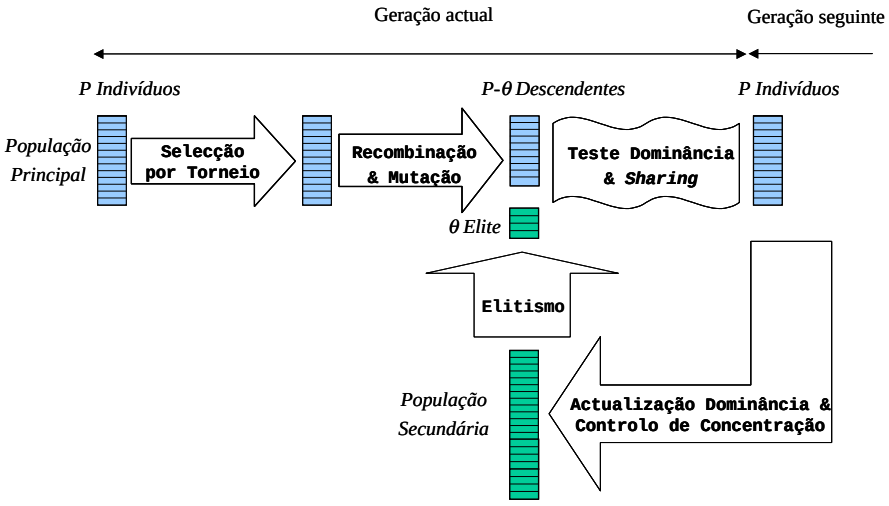

Figura 6.2: Aspecto Geral do *Multiobjective Elitist Genetic Algorithm*

Para evitar este aumento, pode-se impor um tamanho máximo (S) da PS. Em todas as gerações, uma solução não dominada encontrada na população principal x_{nd} é guardada na PS se:

1. todas as soluções de PS forem diferentes de x_{nd};

2. nenhuma das soluções na PS domina x_{nd}.

Após a inserção de uma solução na PS, todas as soluções na PS que se tornam dominadas são eliminadas.

A Figura 6.2 pretende descrever a estrutura e o funcionamento do MEGA. A partir da população principal de P indivíduos são seleccionados por torneio $P - \theta$ indivíduos. Uma nova população é formada pelos $P - \theta$ descendentes gerados por recombinação e mutação, e θ indivíduos seleccionados aleatoriamente a partir da PS (a elite). Em seguida, é feita a avaliação dos indivíduos através do teste de dominância e do mecanismo de *sharing* atrás descritos. Finalmente, as novas soluções não dominadas encontradas são processadas de acordo com o esquema de actualização da PS.

Para ilustrar este algoritmo, considere-se o seguinte problema multi-objectivo:

$$\begin{aligned} \min f_1(x_1, x_2) = &\quad (x_1 + x_2)^2 \\ \min f_2(x_1, x_2) = &\quad (x_1 x_2 - 2)^2. \end{aligned} \tag{6.2.1}$$

As duas variáveis foram representadas por sequências de 32 dígitos binários. Na experiência considerou-se uma população principal de tamanho 100, uma selecção por torneio (com 2 indivíduos), uma recombinação em dois pontos com probabilidade de 0.7, uma mutação uniforme com probabilidade de 0.001 e um valor de σ_{share} de 0.05.

Para fins ilustrativos da importância da PS, a Figura 6.2 mostra as soluções não dominadas (potenciais soluções óptimas de Pareto), após 200 gerações, na população principal e na população secundária para o problema indicado em (6.2.1). Nesta figura é possível verificar-se que um número significativo de soluções não dominadas na população principal é dominada pelas soluções da PS. Isto salienta a importância da utilização de uma PS para evitar a perda de soluções não dominadas que permitem aproximar melhor as soluções óptimas de Pareto. Para além disso, a distribuição de soluções na população principal, no espaço dos objectivos, é claramente menos uniforme.

6.3 Controlo do Tamanho da População Secundária

Como foi atrás referido, à medida que o tamanho da PS aumenta, o tempo de execução e as necessidades de memória também aumentam. Por isso, é conveniente manter tamanhos da PS relativamente pequenos. Assim, o algoritmo anterior pode ser modificado, e, um novo parâmetro d é introduzido, representando a mínima distância desejável entre soluções na PS. Logo, o algoritmo é modificado pela introdução do seguinte passo:

3. a distância de x_{nd} a qualquer uma das soluções na PS que não sejam dominadas por x_{nd} seja superior a d (distância euclidiana medida no espaço dos objectivos).

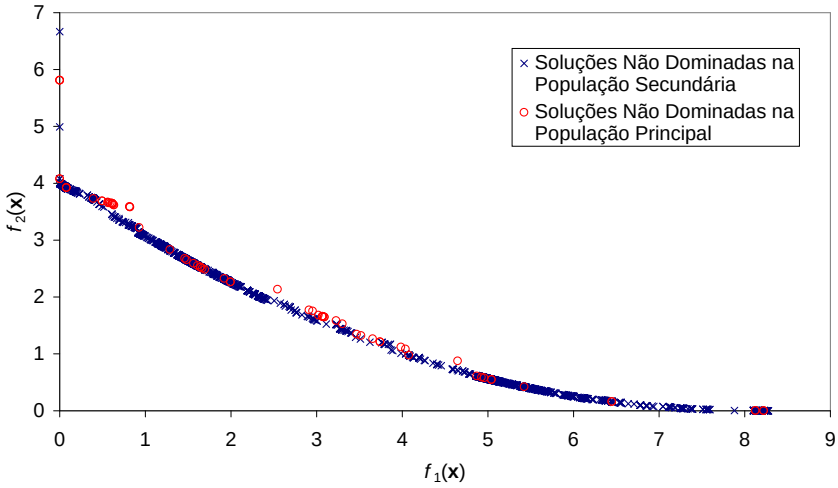

Figura 6.3: Soluções Não Dominadas nas Populações Secundária e Principal

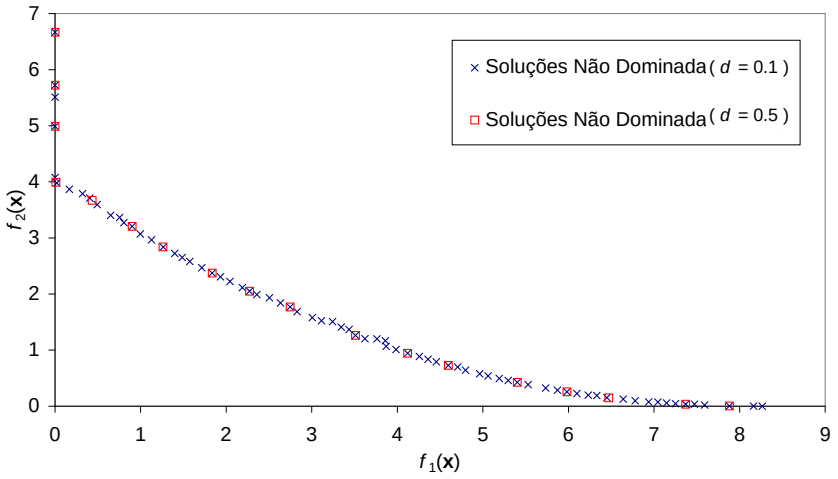

Figura 6.4: Soluções Não Dominadas na População Secundária para d=0.1 e d=0.5

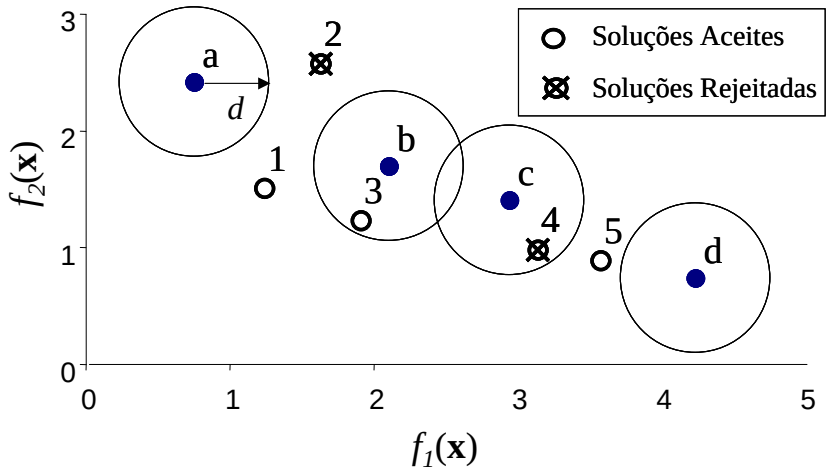

Figura 6.5: Exemplo da Aceitação ou Rejeição de Novas Soluções na População Secundária

A Figura 6.4 apresenta as soluções não dominadas (potenciais soluções óptimas de Pareto) na PS após 200 gerações para $d = 0.1$ e $d = 0.5$. As soluções, nesta figura, são distribuídas de acordo com o valor de d.

A Figura 6.5 ilustra a aceitação ou rejeição de novas soluções na PS. Em cada geração, as soluções não dominadas presentes na população principal são candidatas a serem inseridas na PS. Na Figura 6.5, as soluções **a** a **d** são as soluções presentes na PS (soluções não dominadas encontradas até ao momento). Diversos casos possíveis são ilustrados pelas soluções **1** a **5** a incluir na PS (por simplicidade, todos os casos vão ser considerados individualmente, portanto para todos os casos, a PS conterá as mesmas soluções **a** a **d**). A solução **1** é aceite porque não é dominada por nenhuma das soluções presentes na PS (soluções **a** a **d**) e a distância entre a solução **1** e a solução **a** é superior a d. Depois de se inserir a solução **1** na PS, a solução **b** é removida da PS pois passa a ser dominada pela solução **1**. A solução **2** é obviamente rejeitada porque é dominada pela solução **a**. A solução **3** é aceite porque domina a solução **b** (apesar da distância entre soluções ser inferior a d). As soluções **b** e **c** são removidas da PS depois da inserção da solução **3** na PS porque são dominadas pela solução **3**. No entanto, a solução **4** não é aceite apesar de

ser uma solução não dominada. Isto deve-se ao facto de que a solução **c** não é dominada pela solução **4** e a distância entre a solução **4** e a **c** é inferior a d. Finalmente, a solução não dominada **5** é aceite.

Portanto, o parâmetro d define a região de indiferença em relação à dominância. A ordem pela qual as soluções candidatas são inseridas na PS influencia a aproximação à frente de Pareto. No entanto, esta influência pode ser diminuída por execuções sucessivas com valores decrescentes do parâmetro d, portanto, definindo aproximações à frente de Pareto com aumentos da precisão.

No que diz respeito ao esforço computacional requerido pelo algoritmo, deve-se notar que o número de cálculos da função objectivo é o mesmo com ou sem população secundária e proporcional ao tamanho da população principal (P). A utilização de uma população secundária implica um esforço adicional resultante do teste de dominância das soluções arquivadas. A complexidade do algoritmo sem PS é $\mathcal{O}(sP^2)$ onde s é número de objectivos. Considerando um tamanho limitado da PS (S), o algoritmo deve executar o teste de dominância de complexidade $\mathcal{O}(sSP)$, o que se traduz numa complexidade global de $\mathcal{O}(sSP)$. Esta complexidade depende do tamanho relativo das duas populações, i.e., de quanto o tamanho da população secundária (S) é maior ou menor do que o tamanho da população principal (P). Como exemplo, os tempos de execução para o problema (6.2.1) são 2 seg., 13 seg., 7 min. 42 seg., respectivamente, sem população secundária, com uma população secundária de tamanho limitado igual a 1500 e arquivando todas as soluções não dominadas encontradas ao longo das gerações (com 5000 soluções não dominadas na população secundária).

Tendo em consideração os exemplos e as figuras anteriores, a estratégia elitista descrita com PS, apresenta algumas vantagens quando comparada com os resultados obtidos considerando apenas a população principal:

- o conjunto de todas as soluções não dominadas na PS constitui uma muito melhor

aproximação ao conjunto de soluções óptimas de Pareto;

- as soluções na PS apresentam claramente uma distribuição mais equilibrada ao longo da frente de Pareto;
- o parâmetro d permite a definição da concentração de pontos ao longo da frente de Pareto;
- o tamanho da PS é menor quando comparado com uma abordagem que mantém todas as soluções não dominadas encontradas até ao momento;
- o tempo de computação adicional requerido é negligenciável considerando a qualidade dos resultados quando comparados com os obtidos utilizando apenas a população principal; além disso, o tempo e esforço computacional é muito menor do que o requerido para manter todos os pontos não dominados na PS.

6.4 Resultados Computacionais

O esquema elitista descrito foi testado com alguns problemas multi-objectivo com o objectivo de investigar a influência dos diversos parâmetros.

6.4.1 Casos de Estudo e Medição dos Resultados

Os problemas multi-objectivo foram escolhidos a partir de trabalhos anteriores na área [Sch85, FF98, Pol97, Kur90, ZDT00] como os apresentados na Tabela 5.1 da Secção 5.3. Todos os problemas têm apenas duas funções objectivo e nenhuma restrição.

A métrica aqui considerada para a medição dos resultados é a proposta por Knowles e Corne [KC00] e descrita anteriormente na Secção 5.4. Para todos os resultados apresenta-

Parâmetro	Valor
Número de gerações	250
Tamanho da população	100
Probabilidade de recombinação	0.7
Probabilidade de mutação	$1/b$
σ_{share}	0.027

Tabela 6.1: Parâmetros do *Multiobjective Elitist Genetic Algorithm*

dos, o nível de significância estatística considerado foi de 5% e foram utilizadas 1000 linhas de amostragem.

6.4.2 Parâmetros do Algoritmo Genético Elitista

O MEGA foi aplicado a cada problema fixando-se os parâmetros nos valores que são apresentados na Tabela 6.1 (nenhum esforço foi feito para encontrar os melhores valores dos parâmetros para cada problema).

Para cada problema, o MEGA foi executado 30 vezes e, para cada execução, o conjunto de todas as soluções não dominadas encontradas ao longo da procura foi considerado como o resultado de uma execução de optimização. Independentemente do problema, o critério de paragem foi terminar a execução após 250 gerações. Todas as variáveis de decisão dos problemas foram representadas por uma sequência de 30 dígitos binários. Uma selecção por torneio (com 2 indivíduos), recombinação com dois pontos e mutação uniforme foram adoptadas. A probabilidade de recombinação foi de 0.7 para todos os problemas; enquanto que, a probabilidade de mutação foi de $1/b$, onde b é o comprimento total da sequência de dígitos binários ($b = 30n$ dígitos binários onde n é o número de variáveis de decisão do problema). O valor de σ_{share} foi mantido constante para todos os problemas tal como é indicado na Tabela 6.1. Todas as distâncias entre soluções foram medidas no espaço dos

ZDT1	$\theta = 5$	$\theta = 10$	$\theta = 15$	$\theta = 20$	$\theta = 25$	$\theta = 30$
$\theta = 0$	[90.5,100]	[46.6,100]	[96.0,100]	[71.2,100]	[94.6,100]	[71.6,100]
$\theta = 5$	-	[87.0,100]	[100,99.4]	[98.8,100]	[100,100]	[97.4,92.1]
$\theta = 10$	-	-	[100,40.3]	[100,100]	[100,70.3]	[100,93.8]
$\theta = 15$	-	-	-	[84.3,100]	[100,100]	[79.6,99.9]
$\theta = 20$	-	-	-	-	[84.3,100]	[100,100]
$\theta = 25$	-	-	-	-	-	[96.0,97.2]

Tabela 6.2: Influência do Parâmetro θ (Problema ZDT1, $d = 0$)

objectivos. Tal como se descreve em seguida, diversos cenários foram considerados para estudar a influência dos parâmetros θ (o nível de elitismo) e d.

6.4.3 Influência do elitismo, os parâmetros θ e d

A Tabela 6.2 apresenta os resultados obtidos para o problema ZDT1 com $d = 0$ para diferentes valores de θ. O parâmetro d foi fixado em 0 por forma a garantir que todas as soluções não dominadas encontradas ao longo da procura se encontram no final da execução na PS. Esta tabela mostra que para valores crescentes de θ se verifica uma degradação do desempenho do algoritmo devido à perda de diversidade na população principal. Apesar do parâmetro θ ser o número de soluções que constituem a elite, este pode ser visto como uma percentagem da população principal (por exemplo, considerando uma população principal de 100 indivíduos, se $\theta = 10$, então a elite corresponderá a uma percentagem de 10% da população principal). De forma consistente, os melhores resultados foram obtidos com 10% de elitismo ($\theta = 10$).

O parâmetro d permite controlar a concentração de soluções não dominadas na PS e, consequentemente, conduz a uma boa distribuição das soluções não dominadas no espaço dos objectivos. A Tabela 6.3 apresenta as comparações, par a par, para o AG elitista

ZDT1	$d = 0.014$	$d = 0.027$	$d = 0.054$	$d = 0.108$
$d = 0$	[100,100]	[100,100]	[100,99.3]	[100,59.5]
$d = 0.014$	-	[100,100]	[100,100]	[100,65.1]
$d = 0.027$	-	-	[100,99.9]	[100,73.8]
$d = 0.054$	-	-	-	[100,89.5]

Tabela 6.3: Influência do Parâmetro d (Problema ZDT1, $\theta = 0$)

aplicado ao problema ZDT1. Aqui, valores distintos de d foram considerados ($d = 0$, $d = 0.014$, $d = 0.027$, $d = 0.054$ e $d = 0.108$) e $\theta = 0$. A Figura 6.6 apresenta o número máximo, o número médio e o número mínimo de soluções não dominadas na PS após 250 gerações para valores diferentes de d, para o problema ZDT1.

Para valores de $d = 0.014$ e $d = 0.027$, não existem diferenças significativas nos resultados. Por outro lado, a partir da Figura 6.6 é clara a redução significativa no número médio de soluções não dominadas na PS à medida que d aumenta. Logo, a importância de controlar o valor de d é agora evidente, uma vez que é possível obter um conjunto conveniente de soluções não dominadas com uma redução significativa do esforço para manter a PS, em termos de computação e espaço de memória.

No entanto, é importante notar que o impacto do parâmetro d na definição de um conjunto de soluções não dominadas na PS depende largamente do problema multi-objectivo que se está a resolver e, especialmente, na natureza da frente de soluções óptimas de Pareto do problema. Por isso, o valor de d deve ser ajustado cuidadosamente e, valores próximos ou inferiores aos valores tipicamente sugeridos para o σ_{share} produzem, em geral, bons resultados (quando todas as distâncias são medidas no espaço dos objectivos). A Tabela 6.4 apresenta os resultados para diferentes combinações dos parâmetros θ e d para o problema ZDT1. Tal como anteriormente, é clara a vantagem da utilização de níveis de elitismo convenientes. No entanto, quando é utilizado elitismo, o efeito do parâmetro d é mais efectivo

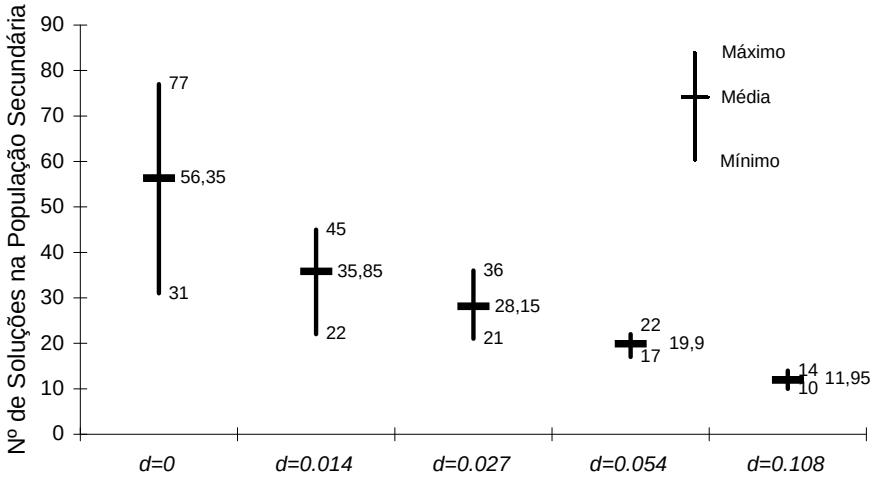

Figura 6.6: Número de Soluções Não Dominadas na População Secundária versus d (Problema ZDT1)

devido ao facto de que as soluções não dominadas da PS são utilizadas para conduzir a procura. Logo, neste caso o valor de d deve ser cuidadosamente escolhido.

A Figura 6.7 apresenta as soluções não dominadas na PS após 250 gerações para o problema ZDT1 com $\theta = 0$ e $\theta = 10$ ($d = 0$). É claro que a melhor aproximação à frente de soluções óptimas de Pareto é atingida com elitismo ($\theta = 10$). Na Figura 6.8, as soluções não dominadas na PS após 250 gerações para o problema ZDT1 com $d = 0$ e $d = 0.014$ ($\theta = 10$) são representadas. Em ambas as figuras são apresentadas as soluções obtidas numa única execução, iniciando a procura a partir da mesma população inicial. O número de soluções não dominadas quando $d = 0$ é superior ao número de soluções não dominadas obtidas para $d = 0.014$ (consequentemente, as soluções não dominadas estão mais concentradas). Um número significativo de soluções para $d = 0$ são dominadas por soluções obtidas para $d = 0.014$. De salientar que a melhor aproximação ao conjunto de soluções óptimas de Pareto obtida para $d = 0.014$ é justificada pela interacção entre os parâmetros d e θ. O parâmetro d, enquanto assegura uma melhor distribuição das soluções na PS, i.e., uma

		$\theta = 0$		$\theta = 10$		
ZDT1		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,100]	[100,100]	[46.6,100]	[51.7,100]	[89.0,100]
	$d = 0.014$	-	[100,100]	[42.9,100]	[49.7,100]	[86.6,100]
	$d = 0.027$	-	-	[38.0,100]	[43.6,100]	[78.2,100]
$\theta = 10$	$d = 0$	-	-	-	[100,100]	[100,87.6]
	$d = 0.014$	-	-	-	-	[100,94.4]

Tabela 6.4: Interacção entre os Parâmetros θ e d (problema ZDT1)

		$\theta = 0$		$\theta = 10$		
SCH		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[91.2,88.6]	[98.0,79.7]	[92.3,94.9]	[44.1,89.0]	[59.7,81.0]
	$d = 0.014$	-	[100,73.1]	[86.0,85.1]	[50.7,99.4]	[69.3,77.0]
	$d = 0.027$	-	-	[75.2,94.2]	[15.7,100]	[52.8,98.4]
$\theta = 10$	$d = 0$	-	-	-	[50.8,90.5]	[68.0,82.0]
	$d = 0.014$	-	-	-	-	[99.5,55.7]

Tabela 6.5: Interacção entre os Parâmetros θ e d (Problema SCH)

maior diversidade, potencia o efeito do elitismo como pode ser visto na Figura 6.8.

As Tabelas 6.5 a 6.12 apresentam os resultados obtidos para diferentes combinações dos parâmetros θ e d nos problemas SCH, FON, POL, KUR, ZDT2, ZDT3, ZDT4 e ZDT6.

A Tabela 6.13 resume os resultados obtidos em todos os problemas para diferentes combinações dos valores dos parâmetros θ e d. Os valores na tabela são as medianas da percentagem do espaço dos objectivos em que, cada combinação de θ e d não é ultrapassada quando comparada com as restantes combinações de θ e d. Novamente, é clara a vantagem do elitismo. De uma forma global, a melhor combinação dos parâmetros θ e d foi $\theta = 10$ e $d = 0.014$.

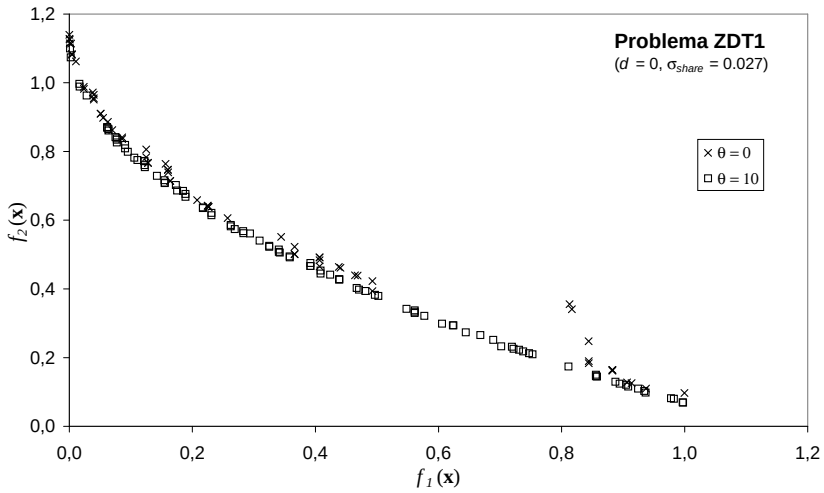

Figura 6.7: Soluções Não Dominadas na População Secundária para $\theta = 0$ e $\theta = 10$ (Problema ZDT1)

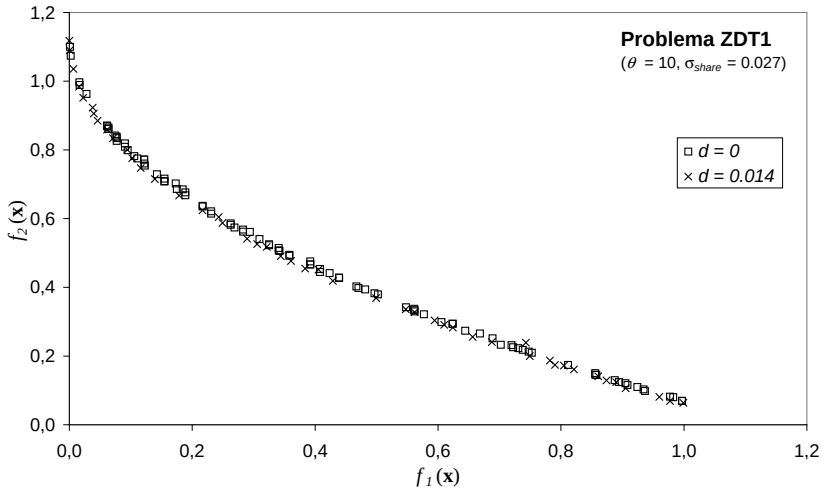

Figura 6.8: Soluções Não Dominadas na População Secundária para $d = 0$ e $d = 0.014$ (Problema ZDT1)

		$\theta = 0$		$\theta = 10$		
FON		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,28.4]	[100,15.7]	[75.8,87.8]	[80.2,43.1]	[94.3,28.3]
	$d = 0.014$	-	[100,38.3]	[7.3,100]	[56.3,96.6]	[87.5,48.7]
	$d = 0.027$	-	-	[5.3,100]	[10.8,100]	[68.5,97.1]
$\theta = 10$	$d = 0$	-	-	-	[93.0,27.9]	[100,13.3]
	$d = 0.014$	-	-	-	-	[100,15.9]

Tabela 6.6: Interacção entre os Parâmetros θ e d (Problema FON)

		$\theta = 0$		$\theta = 10$		
POL		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,94.2]	[100,90.4]	[97.7,77.3]	[75.1,95.9]	[55.8,92.2]
	$d = 0.014$	-	[100,95.0]	[93.0,99.9]	[73.8,99.9]	[55.3,96.4]
	$d = 0.027$	-	-	[86.2,99.9]	[66.7,100]	[51.7,99.9]
$\theta = 10$	$d = 0$	-	-	-	[72.1,95.7]	[74.8,86.5]
	$d = 0.014$	-	-	-	-	[77.8,83.9]

Tabela 6.7: Interacção entre os Parâmetros θ e d (Problema POL)

		$\theta = 0$		$\theta = 10$		
KUR		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,83.6]	[100,69.6]	[84.5,66.8]	[63.6,83.9]	[75.3,75.9]
	$d = 0.014$	-	[100,84.0]	[75.0,74.8]	[55.3,97.9]	[72.6,84.4]
	$d = 0.027$	-	-	[63.7,82.6]	[34.9,100]	[58.1,97.6]
$\theta = 10$	$d = 0$	-	-	-	[54.2,93.7]	[67.3,75.2]
	$d = 0.014$	-	-	-	-	[99.2,68.5]

Tabela 6.8: Interacção entre os Parâmetros θ e d (Problema KUR)

		$\theta = 0$		$\theta = 10$		
ZDT2		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,100]	[100,100]	[100,100]	[100,100]	[100,100]
	$d = 0.014$	-	[100,100]	[100,100]	[100,100]	[99.8,100]
	$d = 0.027$	-	-	[100,100]	[100,100]	[99.4,100]
$\theta = 10$	$d = 0$	-	-	-	[100,100]	[100,100]
	$d = 0.014$	-	-	-	-	[100,100]

Tabela 6.9: Interacção entre os Parâmetros θ e d (Problema ZDT2)

		$\theta = 0$		$\theta = 10$		
ZDT3		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,100]	[100,100]	[76.6,100]	[23.2,100]	[83.4,100]
	$d = 0.014$	-	[100,100]	[74.5,100]	[20.7,100]	[80.8,100]
	$d = 0.027$	-	-	[71.6,100]	[18.6,100]	[77.4,100]
$\theta = 10$	$d = 0$	-	-	-	[52.3,100]	[99.5,95.2]
	$d = 0.014$	-	-	-	-	[100,78.2]

Tabela 6.10: Interacção entre os Parâmetros θ e d (Problema ZDT3)

		$\theta = 0$		$\theta = 10$		
ZDT4		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,100]	[100,100]	[100,100]	[100,100]	[100,100]
	$d = 0.014$	-	[100,100]	[100,100]	[100,100]	[99.9,100]
	$d = 0.027$	-	-	[100,100]	[100,100]	[100,100]
$\theta = 10$	$d = 0$	-	-	-	[100,100]	[100,100]
	$d = 0.014$	-	-	-	-	[100,100]

Tabela 6.11: Interacção entre os Parâmetros θ e d (Problema ZDT4)

		$\theta = 0$		$\theta = 10$		
ZDT6		$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
$\theta = 0$	$d = 0$	[100,100]	[100,100]	[32.5,100]	[30.4,100]	[38.1,100]
	$d = 0.014$	-	[100,100]	[32.5,100]	[30.4,100]	[38.1,100]
	$d = 0.027$	-	-	[37.0,100]	[35.1,100]	[37.9,100]
$\theta = 10$	$d = 0$	-	-	-	[100,100]	[100,100]
	$d = 0.014$	-	-	-	-	[100,100]

Tabela 6.12: Interacção entre os Parâmetros θ e d (Problema ZDT6)

	$\theta = 0$			$\theta = 10$		
Problema	$d = 0$	$d = 0.014$	$d = 0.027$	$d = 0$	$d = 0.014$	$d = 0.027$
SCH	91.2	85.1	94.2	85.1	99.4	81.0
FON	94.3	56.3	15.7	87.8	96.6	28.3
POL	97.7	93.0	86.2	77.3	95.9	92.2
KUR	84.5	75.0	63.7	67.3	97.9	75.9
ZDT1	89.0	86.6	78.2	100.0	100.0	100.0
ZDT2	100.0	100.0	100.0	100.0	100.0	100.0
ZDT3	83.4	80.8	77.4	100.0	100.0	100.0
ZDT4	100.0	100.0	100.0	100.0	100.0	100.0
ZDT6	38.1	38.1	37.9	100.0	100.0	100.0

Tabela 6.13: Desempenho Global

6.5 Discussão dos Resultados

Em problemas multi-objectivo reais as soluções óptimas de Pareto não são, em geral, conhecidas à partida. Portanto, qualquer algoritmo projectado para encontrar este conjunto de soluções deve assegurar não só uma distribuição uniforme das soluções ao longo da frente de Pareto, mas também, evitar convergência prematura.

O algoritmo apresentado, MEGA (*Multiobjective Elitist Genetic Algorithm*), foi testado em diversos problemas, produzindo uma boa distribuição das soluções ao longo da frente de soluções óptimas de Pareto. Além disso, o nível de elitismo e a concentração de pontos pode ser controlada por dois parâmetros e, para valores razoáveis, o aumento no tempo computacional é negligenciável.

O elitismo provou ser muito útil na obtenção de uma boa aproximação ao conjunto de soluções óptimas de Pareto. No entanto, o controlo do elitismo requer um maior estudo, e este trabalho mostra como o nível de elitismo pode influenciar o desempenho dos algoritmos. Para além disso, um compromisso entre uma população secundária grande e uma boa distribuição das soluções pode ser atingida utilizando um parâmetro que controla a densidade de soluções na aproximação ao conjunto de soluções óptimas de Pareto.

Capítulo 7

Uma Estratégia Evolutiva Elitista

Durante a década passada, a grande maioria das abordagens evolucionárias a problemas de optimização multi-objectivo foi baseada em AGs [Sch85, FF93, HNG94, SD94, ZT99], contudo implementações baseadas em EEs [Rec94] são muito raras, como os algoritmos propostos por Kursawe [Kur90], e Knowles e Corne [KC00]. No entanto, esta última abordagem não contempla algumas das características tradicionais das EEs, tais como, a codificação real das variáveis de decisão e a adaptação dos tamanhos do passo para a mutação. Por isso, é crucial investigar como estender as EEs para optimização multi-objectivo, uma vez que, no passado, provaram ser algoritmos poderosos para optimização uni-objectivo. Assim, uma nova abordagem à optimização multi-objectivo, baseada em EEs, foi desenvolvida [CO02b]. No novo algoritmo, *Multiobjective Elitist Evolution Strategy* (MEES), foi feito um esforço para manter as principais características das EEs utilizadas em optimização uni-objectivo. Em particular, aspectos como a representação real das variáveis de decisão, em conjunto com a adaptação dos tamanhos do passo na mutação e a recombinação foram adoptados. Vários mecanismos, como o elitismo, foram introduzidos para melhorar o desempenho do algoritmo, como previamente foi sugerido por Zitzler *et al.* [ZDT00] e Van Veldhuizen e Lamont [VVL00]. Um novo esquema de adaptação do parâmetro de

sharing bem como um esquema de selecção geométrica provaram ser úteis na procura.

7.1 Medição do Desempenho

O algoritmo proposto mantém as principais características das EEs e introduz novos mecanismos como o elitismo e *sharing* para melhorar o seu desempenho em optimização multi-objectivo tais como a adaptação dos tamanhos dos deslocamentos e a representação real das variáveis de decisão.

A Figura 7.1 ilustra a estrutura e o funcionamento do MEES. Esta abordagem difere das EEs convencionais no que diz respeito ao operador de selecção que dá ênfase à não dominância das soluções. Em cada geração a dominância é testada na fase de selecção, definindo uma aproximação ao conjunto de soluções óptimas de Pareto. Por outro lado, um método de *sharing* é utilizado para distribuir as soluções da população ao longo da região das soluções óptimas de Pareto. A selecção determinística usual foi modificada para permitir tratar problemas de optimização multi-objectivo. A representação real das variáveis de decisão, e os operadores de mutação e recombinação permaneceram como os usuais. Os tamanhos do passo para a mutação foram adaptados de acordo com um esquema não isotrópico como o expresso pela equação (3.1.12).

Em cada geração, todas as soluções não dominadas das λ ou $\mu + \lambda$ soluções formam a primeira frente. A estas soluções é atribuída uma medida do desempenho de 1. Para manter diversidade, um esquema de *sharing* é então aplicado aos valores das medidas de desempenho destas soluções [DG89]. Portanto, o valor de medida do desempenho é multiplicado por uma quantidade proporcional ao número de soluções que distam de uma distância inferior a um parâmetro, o σ_{share}. Todas as distâncias são medidas no espaço dos objectivos. A partir daqui, as soluções da primeira frente são ignoradas temporariamente, e o resto das soluções são processadas. Para o segundo nível de soluções não dominadas é

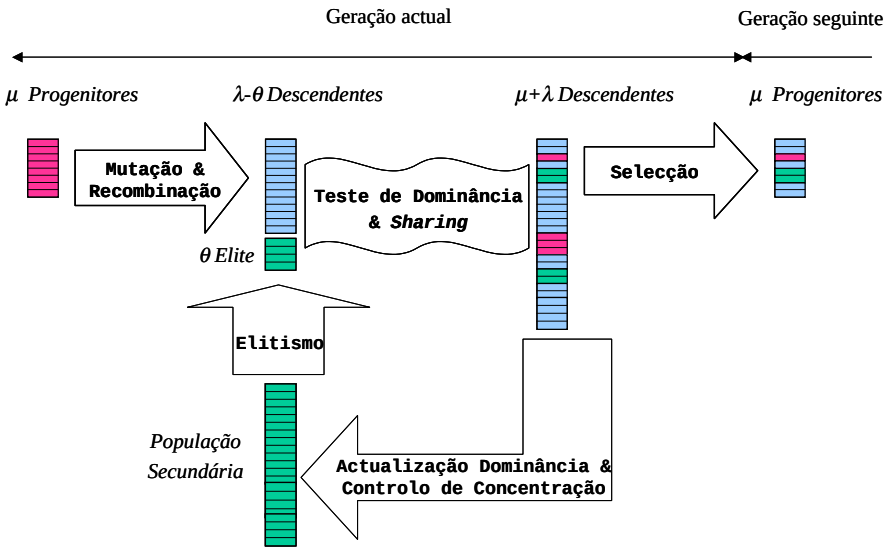

Figura 7.1: Aspecto Geral do *Multiobjective Elitist Evolution Strategy*

atribuído um valor igual ao maior valor de medida do desempenho das soluções da primeira frente mais 1. Em seguida, para cada solução, este valor é multiplicado por uma quantidade proporcional ao número de soluções que distam de uma distância inferior a σ_{share}. Este processo é repetido até que todas as λ ou $\mu + \lambda$ soluções tenham um valor de medida do desempenho atribuído. Este esquema garante que qualquer solução da frente $k + 1$ apresenta uma medida do desempenho superior a qualquer solução da frente k.

7.1.1 Adaptação do Parâmetro de *Sharing*

O funcionamento do mecanismo de *sharing* depende principalmente do parâmetro σ_{share}. No entanto, a escolha do valor para este parâmetro não é fácil. Este problema foi identificado e discutido por Deb e Goldberg [DG89] e Fonseca e Fleming [FF93]. Em particular, alguns esquemas adaptativos foram propostos por Fonseca e Fleming [FF98]. Uma vez que o valor deste parâmetro deve ser escolhido cuidadosamente para se obter um conjunto de aproximações às soluções óptimas de Pareto bem distribuídas no espaço dos objectivos, um

novo esquema adaptativo do parâmetro de *sharing* foi desenvolvido para integrar o MEES.

O parâmetro σ_{share} é a distância máxima entre soluções necessária para formar uma frente bem distribuída. No entanto, este parâmetro depende do número desejável de soluções distintas na frente e dos limites superior e inferior do espaço das soluções. A comparação das soluções pode ser feita quer no espaço das variáveis de decisão quer no espaço dos objectivos. É possível existirem diferentes soluções no espaço das variáveis (correspondendo a diferentes vectores de variáveis de decisão) com os mesmos valores das funções objectivo. No entanto, muitas vezes, a diversidade e a distribuição uniforme das soluções no espaço das objectivos (aproximando a frente óptima de Pareto) são de principal importância. Por outro lado, a comparação das soluções ao nível do espaço das variáveis, quando existem múltiplas soluções com os mesmos valores das funções objectivo, poderá tender a impedir que a procura se faça em determinadas regiões do espaço dos objectivos.

Uma vez que as soluções na população são ordenadas em termos da dominância, definindo diversas frentes (Figura 7.1), é conveniente considerar diferentes valores de σ_{share} de acordo com o número de soluções em cada frente. Portanto, os valores particulares de σ_{share}, calculados para cada uma das k frentes, devem ter em conta:

- a distribuição das soluções da frente no espaço dos objectivos;
- o número de soluções da frente n_k.

Cada frente k tem n_k soluções e os limites inferiores, f_1^*, f_2^* e f_s^*, e limites superiores, F_1^*, F_2^* e F_s^*, para os s objectivos são conhecidos. Logo, a *solução ideal*, $f_{ideal}^* = (f_1^*, f_2^*, \ldots, f_s^*)$ e a *solução nadir*, $F_{nadir}^* = (F_1^*, F_2^*, \ldots, F_s^*)$, podem ser calculadas (Figura 7.2). A distância euclidiana entre estas duas soluções pode ser calculada da seguinte forma:

$$d_{norm} = \sqrt{\sum_{i=1}^{s} (f_i^* - F_i^*)^2} \tag{7.1.1}$$

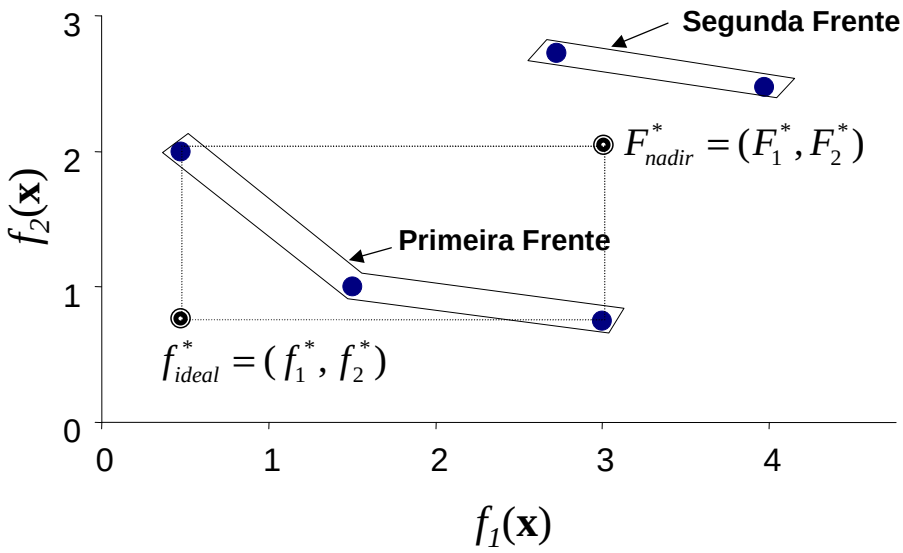

Figura 7.2: Soluções Extremas do Espaço dos Objectivos

Como primeira abordagem, para estimar o valor do parâmetro σ_{share}, esta distância pode ser dividida em $n_k - 1$ partes. O valor estimado para σ_{share}, σ'_{share}, será metade desta quantidade (o raio) e é dado por:

$$\sigma'_{share} = \begin{cases} \frac{d_{norm}}{2\sqrt[s-1]{n_k - 1}} & \text{se} \quad n_k > 1 \\ 0 & \text{se} \quad n_k = 1 \end{cases} \tag{7.1.2}$$

É também possível definir as distâncias entre as soluções extremas de cada objectivo, $\overline{f}_1 = (f_1^*, F_2^*, \ldots, F_s^*)$, $\overline{f}_2 = (F_1^*, f_2^*, \ldots, F_s^*)$ e $\overline{f}_s = (F_1^*, F_2^*, \ldots, f_s^*)$, à *solução ideal*:

$$d_{j,ideal} = \sqrt{\sum_{i=1, i\neq j}^{s} (f_i^* - F_i^*)^2} \tag{7.1.3}$$

As distância a partir das soluções extremas e a *solução nadir* são dadas por:

$$d_{j,nadir} = d_{m-j,ideal} = \sqrt{\sum_{i=1, i\neq s-j}^{s} (f_i^* - F_i^*)^2} \tag{7.1.4}$$

As distâncias entre os pontos extremos e a *solução ideal* podem ser utilizadas para calcular outra estimativa de σ_{share}. Esta estimativa extrema é mais forte do que a dada

por (7.1.1), uma vez que tende a reduzir a distância entre soluções no espaço dos objectivos, e, para $n_k > 1$ (se $n_k = 1$ então $\sigma''_{share} = 0$), é dada por:

$$\sigma''_{share} = \frac{\sum_{j=1}^{s} d_{j,ideal}}{2\sqrt[s-1]{n_k - 1}} \tag{7.1.5}$$

Estas duas estimativas, σ'_{share} e σ''_{share}, podem ser combinadas como:

$$\sigma_{share} = \alpha\sigma'_{share} + (1-\alpha)\sigma''_{share} = \frac{\alpha d_{norm} + (1-\alpha)\sum_{j=1}^{s} d_{j,ideal}}{2\sqrt[s-1]{n_k - 1}} \tag{7.1.6}$$

onde α é um parâmetro. Se α for 1 ou 0, então a distância entre soluções no espaço dos objectivos tende a aumentar ou a diminuir. Com valores intermédios de α, valores compromisso de σ_{share} são obtidos.

7.1.2 Selecção

Na forma mais simples, em cada geração, apenas μ das λ ou $\mu+\lambda$ soluções são seleccionadas para a geração seguinte. Duas situações são consideradas:

- se o número de soluções na primeira frente, n_1, não for superior a μ, então é aplicada uma selecção determinística;
- caso contrário, se n_1 for maior do que μ, então é aplicada uma selecção por torneio.

A selecção determinística consiste em, após ordenar os λ ou $\mu+\lambda$ indivíduos de acordo com a sua medida do desempenho, seleccionar os μ melhores (os que apresentam menor valor da medida do desempenho). Obviamente, esta selecção é semelhante à selecção tradicional das EEs, no sentido que apenas os melhores indivíduos estarão presentes na geração seguinte. Por outro lado, quando o número de soluções na primeira frente é grande (maior que μ) então é adoptado um esquema de selecção que garante que todas as soluções não dominadas têm uma possibilidade de estarem presentes na geração seguinte. Esta selecção consiste em, após a ordenação dos λ ou $\mu+\lambda$ indivíduos, fazer um torneio entre as soluções da primeira

frente. O torneio consiste em escolher aleatoriamente dois indivíduos e seleccionar o melhor deles (o que tiver menor valor da medida do desempenho). No entanto, esta selecção tem as seguintes principais desvantagens:

- quando $n_1 \leq \mu$, o mecanismo de *sharing* não tem efeito real nas soluções da primeira frente;
- não é possível controlar a pressão de selecção para preservar, por exemplo, diversidade durante a procura.

Um novo mecanismo de selecção pode ser desenvolvido por forma a minimizar estes inconvenientes. A ideia principal é, para cada frente, seleccionar quantidades distintas de soluções para a geração seguinte. As quantidades de soluções para cada frente são escolhidas de tal forma que o número de soluções seleccionadas da primeira frente seja maior que o número de soluções da segunda frente, e assim por diante. Se se considerar uma progressão geométrica, então a quantidade de soluções Q_k para a frente k é dada por

$$Q_k = \mu \frac{r^{K-k}}{\left(\frac{1-r^K}{1-r}\right)} = \mu(1-r)\frac{r^{K-k}}{1-r^K} \tag{7.1.7}$$

onde K é o número total de frentes e r é a razão de variação de Q_k entre frentes (se r for igual a 2 então Q_1 será aproximadamente o dobro de Q_2 e assim por diante). De notar que, para todo o $K \geq 2$, se $r > 1$ então é garantido que $Q_1 > Q_2$. Mais genericamente, para todo o $1 < k < K-1$, se $r > 1$ então $Q_k \geq Q_{k+1}$. Os valores de Q_k para $r = 2$ e considerando valores diferentes de K são apresentados na Tabela 7.1.

No entanto, é importante adaptar o valor de r durante a procura. No início da procura, os valores de n_1 são, em geral, pequenos, por isso, se r for mantido constante, é possível que o valor calculado para Q_1 se torne inadequadamente grande. Logo, é possível adaptar o valor de r da seguinte forma:

$$r = 1 + (ps - 1)\frac{n_1}{\mu} \tag{7.1.8}$$

Q_k		K									
		1	2	3	4	5	6	7	8	9	10
	1	100	67	57	53	52	51	50	50	50	50
	2		33	29	27	26	25	25	25	25	25
	3			14	13	13	13	13	13	13	13
	4				7	6	6	6	6	6	6
k	5					3	3	3	3	3	3
	6						2	2	2	2	2
	7							1	1	1	1
	8								0	0	0
	9									0	0
	10										0

Tabela 7.1: Variação de Q_k para $r = 2$

onde ps ($ps > 1$) é um parâmetro que representa a pressão de selecção. É claro que, se $n_1 \simeq \mu$ então $Q_1 \simeq psQ_2$. Quando n_1 é pequeno comparado com μ então o valor de r é aproximadamente igual a 1. Esta é uma situação comum no início da procura. À medida que n_1 aumenta ao longo da procura, o valor de r também aumenta, tendendo para o valor de ps quando $n_1 \simeq \mu$. Se $n_1 > \mu$ então $r > ps$ e Q_1 irá tender para μ.

Depois do cálculo de r, as quantidades Q_k são calculadas para todas as frentes. Então, duas situações são consideradas:

- se o número de soluções na frente k, n_k, não for superior a Q_k, então uma selecção determinística é aplicada, i.e., todas as n_k soluções da frente k são seleccionadas;
- Caso contrário, se n_k for maior do que Q_k, então é feita uma selecção por torneio entre as soluções da frente k.

Como anteriormente foi descrito, a selecção determinística consiste em após a ordenação dos λ ou $\mu + \lambda$ indivíduos de acordo com os valores da medida do desempenho, seleccionar os μ melhores (os que tiverem menor valor da medida do desempenho). No entanto, uma vez que n_k é inferior a Q_k, apenas n_k soluções são seleccionadas. Neste caso, as soluções restantes $Q_k - n_k$ são acrescentadas à quantidades de soluções a seleccionar para a frente seguinte. Portanto, para a frente $k+1$, a quantidade de soluções efectivamente considerada Q'_{k+1} será $Q_{k+1} + (Q_k - n_k)$, se $Q_k - n_k > 0$ e Q_{k+1}, caso contrário. Se n_k for maior do que Q_k, um esquema de selecção por torneio é utilizado para seleccionar, de cada frente k, a quantidade Q_k de soluções para a geração seguinte. Como anteriormente, este torneio consiste em escolher duas soluções a partir dos λ ou $\mu + \lambda$ indivíduos e seleccionar a melhor. Este esquema de selecção é ilustrado na Tabela 7.2. Para a situação considerada nesta tabela, o objectivo é seleccionar 100 soluções das 200 soluções presentes em 10 frentes. Tomando-se r igual a 2, o número de soluções para as frentes Q_k é dado pela última coluna da Tabela 7.1.

Este esquema de selecção, permitindo controlar a pressão de selecção, será referida como selecção $+_{ps}$ ou $,_{ps}$. O esquema de selecção mais simples descrito no início desta secção será referido como selecção + ou ,. De notar que, quando ps é maior que μ, os esquemas de selecção $+_{ps}$ e $,_{ps}$ tendem a ter um desempenho semelhante aos esquemas de selecção + e ,.

Deb e Goel [DG01] sugeriram um esquema de selecção para controlar o nível de elitismo utilizando um parâmetro fixo. Os resultados da aplicação do NSGA-II com elitismo controlado a um conjunto de problemas multi-objectivo indicou uma propriedade de convergência superior ao NSGA-II original. No entanto, nesta abordagem o valor de r é mantido constante ao longo da procura. No início da procura, o valor de r pode ser inadequadamente alto. Os esquemas de selecção $+_{ps}$ e $,_{ps}$ evitam este inconveniente através da adaptação deste parâmetro ao longo da procura de acordo com o número de soluções na primeira

k	n_k	Q_k	$Q_k - n_k$	Q'_k	Comentário
1	85	50	< 0	50	50 soluções são seleccionadas das 85 por torneio
2	50	25	< 0	25	25 soluções são seleccionadas das 50 por torneio
3	12	13	1	12	12 soluções são seleccionadas deterministicamente
4	15	6	< 0	7	7 soluções são seleccionadas das 15 por torneio
5	12	3	< 0	3	3 soluções são seleccionadas das 12 por torneio
6	9	2	< 0	2	2 soluções são seleccionadas das 9 por torneio
7	8	1	< 0	1	1 solução é seleccionada das 8 por torneio
8	5	0	< 0	0	nenhuma solução é seleccionada
9	3	0	< 0	0	nenhuma solução é seleccionada
10	1	0	< 0	0	nenhuma solução é seleccionada

Tabela 7.2: Exemplo de Selecção $+_{ps}$ para $r = 2$

frente. Este esquema adaptativo pretende prevenir a convergência prematura.

7.1.3 Esquema Elitista

A técnica elitista é baseada numa população secundária (PS) tal como foi descrito na Secção 6.2. O mesmo esquema de controlo do tamanho da PS é utilizado (ver Secção 6.3).

Um parâmetro θ é utilizado para controlar o nível de elitismo. Este parâmetro representa o número máximo de soluções não dominadas da PS, a elite, que são introduzidas na população principal. Se o número de soluções da PS (n_{PS}) for maior que θ, então θ soluções não dominadas são aleatoriamente seleccionadas da PS para formar a elite. Caso contrário, apenas n_{PS} soluções não dominadas são seleccionadas da PS para formar a elite. Neste último caso, a elite terá apenas n_{PS} membros. Um parâmetro d é utilizado para definir a concentração de soluções na PS.

O MEES utiliza um população de μ progenitores e, em todas as gerações, uma po-

pulação de λ descendentes é gerada. Para além disso, uma população secundária arquivando soluções não dominadas é mantida. Se se considerar uma EE-$(\mu/\rho+\lambda)$, então, em cada geração, uma vez que o teste de dominância é aplicado a $\mu+\lambda$ indivíduos, o número de comparações requerido é $\mathcal{O}(s(\mu+\lambda)^3)$. O mecanismo de *sharing* (no espaço dos objectivos) aplicado a $\mu+\lambda$ indivíduos das populações de progenitores e de descendentes requer, no pior dos casos, $\mathcal{O}(s(\mu+\lambda)^2)$ comparações. Assumindo-se um tamanho máximo da PS igual a S, a actualização da PS implica $\mathcal{O}(sS(\mu+\lambda))$ comparações, uma vez que cada nova solução não dominada encontrada nas populações de progenitores ou de descendentes (no pior dos casos $\mu+\lambda$ indivíduos) é comparada, em termos de dominância, com todas as soluções da PS. Se se assumir que $S \simeq \mu+\lambda$, então são requeridas $\mathcal{O}(s(\mu+\lambda)^2)$ avaliações. A complexidade global do algoritmo é dada por $\mathcal{O}(s(\mu+\lambda)^3)+\mathcal{O}(s(\mu+\lambda)^2)+\mathcal{O}(s(\mu+\lambda)^2)$, i.e., $\mathcal{O}(s(\mu+\lambda)^3)$.

7.2 Resultados Computacionais

Diversas experiências foram feitas com o objectivo de, por um lado, investigar a influência dos parâmetros no desempenho do algoritmo e, por outro lado, comparar o algoritmo com outras abordagens evolucionárias.

7.2.1 Casos de Estudo e Medição dos Resultados

Os problemas multi-objectivo aqui considerados foram escolhidos a partir dos trabalhos de Zitzler *et al.* [ZDT00] (ver Tabela 5.1 da Secção 5.3). Todos os problemas têm duas funções objectivo e nenhuma restrição.

A métrica aqui considerada para medição dos resultados é a proposta por [KC00] e descrita anteriormente na Secção 5.4. Para todos os resultados apresentados, o nível de significância estatística considerado foi de 5% e foram utilizadas 1000 linhas de amostragem.

7.2.2 Parâmetros da Estratégia Evolutiva Elitista

O MEES foi aplicado a cada problema com o mesmo conjunto de valores para os parâmetros (nenhum esforço foi feito para encontrar os melhores valores dos parâmetros para cada problema). Os valores iniciais para os desvios padrão (tamanhos do passo) e parâmetros para a sua adaptação durante a procura foram os sugeridos para as EEs em optimização uni-objectivo; i.e., os valores iniciais para σ_i foram dados pela equação (3.1.1) com $\Delta x_k = (\overline{x_k} - \underline{x_k})/2$ (onde $\overline{x_k}$ e $\underline{x_k}$ são os limites superior e inferior da variável x_k com $k = 1, \ldots, n$) e $\Delta\sigma = \Delta\sigma' = 1/\sqrt{n}$. Os pontos da população inicial foram gerados aleatoriamente. Diversos cenários foram considerados para estudar os efeitos do operador de recombinação, o mecanismo de selecção, o elitismo e o parâmetro d. Logo, para cada cenário, todos os valores dos parâmetros foram mantidos constantes excepto os do que estava a ser estudado (a interacção entre parâmetros não foi estudada).

7.2.3 Influência da Recombinação

O MEES sem recombinação parece ter dificuldades em obter uma boa distribuição das soluções não dominadas quando aplicado a problemas multi-objectivo com um grande número de variáveis de decisão. Isto é ilustrado na Figura 7.3, que representa as soluções não dominadas obtidas, numa única execução, para o problema ZDT2 com 5 e 30 variáveis de decisão para uma EE-(100+150) sem recombinação e, com σ_{share}=0.027, $d = 0$ e $\theta = 0$. O critério de paragem foi terminar a execução após 250 gerações. É evidente que foi obtida uma boa aproximação ao conjunto de soluções óptimas de Pareto para o problema ZDT2 com 5 variáveis. No entanto, para 30 variáveis, os resultados são fracos, uma vez que, as soluções obtidas estão longe da frente de soluções óptimas de Pareto, e não estão distribuídas uniformemente no espaço dos objectivos.

Uma vez que o MEES sem recombinação apresenta um desempenho fraco para proble-

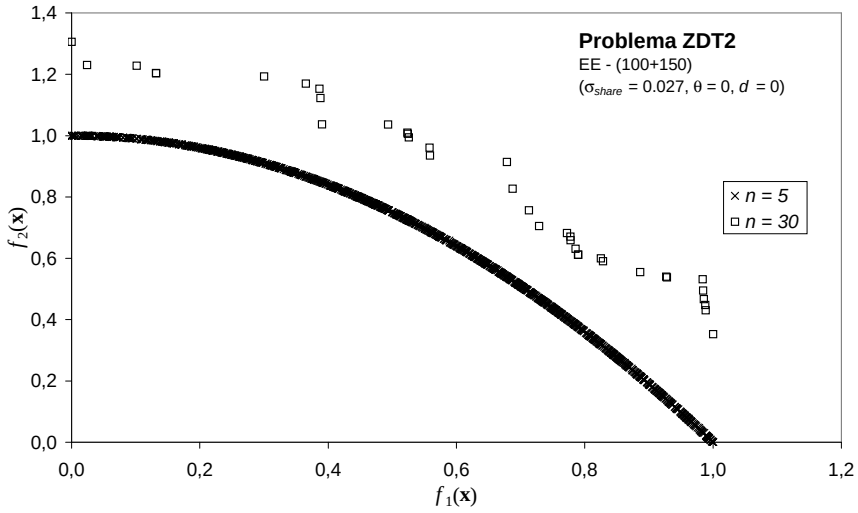

Figura 7.3: Resultados para o Problema ZDT2 com 5 e 30 Variáveis de Decisão

mas multi-objectivo com um grande número de variáveis de decisão, diversos cenários do MEES com recombinação foram testados. Foram considerados cenários que combinam os esquemas de recombinação mais usuais:

- sem nenhuma recombinação (cenário NOrec);
- recombinação intermédia nas variáveis de decisão e nos desvios padrão (cenário IIrec);
- recombinação intermédia nas variáveis de decisão e recombinação discreta nos desvios padrão (cenário IDrec);
- recombinação discreta nas variáveis de decisão e recombinação intermédia nos desvios padrão (cenário DIrec);
- recombinação discreta nas variáveis de decisão e nos desvios padrão (cenário DDrec).

Para os cenários com recombinação, uma EE-(100/100+150) foi aplicada (obviamente, uma EE-(100+150) foi considerada para o cenário sem recombinação) com σ_{share}=0.027,

ZDT1	IIrec	IDrec	DIrec	DDrec
NOrec	[100,3.8]	[100,2.2]	[2.7,100]	[2.7,100]
IIrec	-	[100,7.2]	[3.6,100]	[3.7,100]
IDrec	-	-	[2.1,100]	[2.1,100]
DIrec	-	-	-	[56.7,100]

Tabela 7.3: Influência da Recombinação (Problema ZDT1)

$d = 0$ e $\theta = 0$. Tal como anteriormente, o critério de paragem foi terminar a execução após 250 gerações. Para cada cenário, o MEES foi executado 30 vezes. A Tabela 7.3 apresenta os resultados obtidos em todos os cenários para o problema ZDT1. Todos os cenários foram comparados aos pares usando a técnica estatística anteriormente descrita. É evidente que os melhores resultados foram obtidos para o cenário DDrec, i.e., quando a recombinação discreta é aplicada nas variáveis de decisão e nos desvios padrão.

Os resultados obtidos pelo MEES com recombinação discreta nas variáveis de decisão e nos desvios padrão (uma EE-(100/100+150)) podem ser comparados, para o problema ZDT2, com os obtidos pelo MEES sem recombinação. A comparação é ilustrada pela Figura 7.4, que representa as soluções não dominadas obtidas numa única execução após 250 gerações. A aproximação ao conjunto de soluções óptimas de Pareto obtida pelo MEES com recombinação é muito melhor do que a obtida pelo MEES sem recombinação.

7.2.4 Influência do Elitismo

Com o objectivo de estudar a influência do nível de elitismo, uma EE-(100/100+150) com recombinação discreta nas variáveis de decisão e nos desvios padrão foi aplicada ao problema ZDT1. Considerou-se os mesmos valores para os parâmetros com excepção de θ, que foi variado de 0 a 100 como é apresentado na Tabela 7.4. O parâmetro d foi fixado em

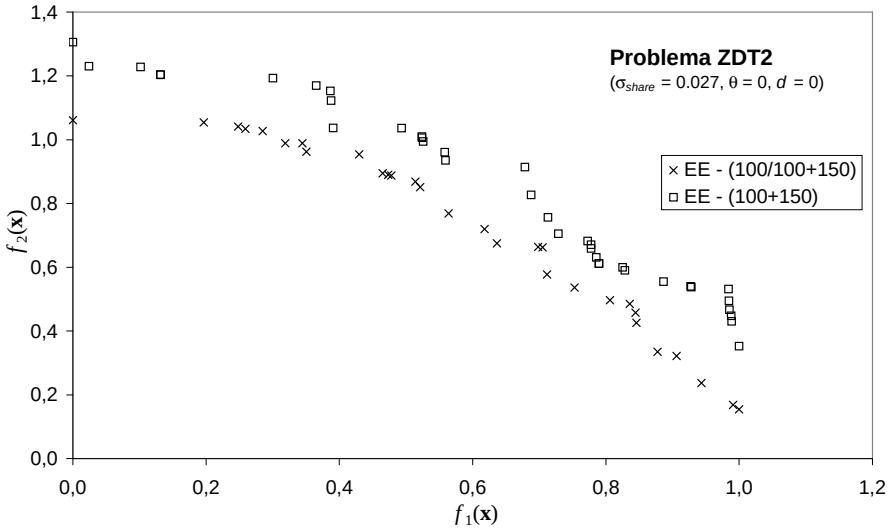

Figura 7.4: Resultados com e sem Recombinação (Problema ZDT2)

ZDT1	$\theta = 10$	$\theta = 20$	$\theta = 50$	$\theta = 100$
$\theta = 0$	[0.3,100]	[0.3,100]	[0.3,100]	[0.3,100]
$\theta = 10$	-	[100,32.1]	[100,13.5]	[100,13.5]
$\theta = 20$	-	-	[96.9,83.9]	[96.9,83.9]
$\theta = 50$	-	-	-	[100,100]

Tabela 7.4: Influência do Elitismo (Problema ZDT1)

0 por forma a garantir que na PS todas as soluções não dominadas encontradas durante a procura estão presentes. Esta tabela mostra que para maiores valores de θ existe uma degradação do desempenho do algoritmo, devido à perda de diversidade na população principal. No entanto, também é claro que os melhores resultados foram obtidos com elitismo. Mais precisamente, os melhores resultados foram obtidos com $\theta = 10$.

A comparação entre diferentes níveis de elitismo é ilustrada pela Figura 7.5, que representa as soluções não dominadas obtidas numa única execução, após 250 gerações para o

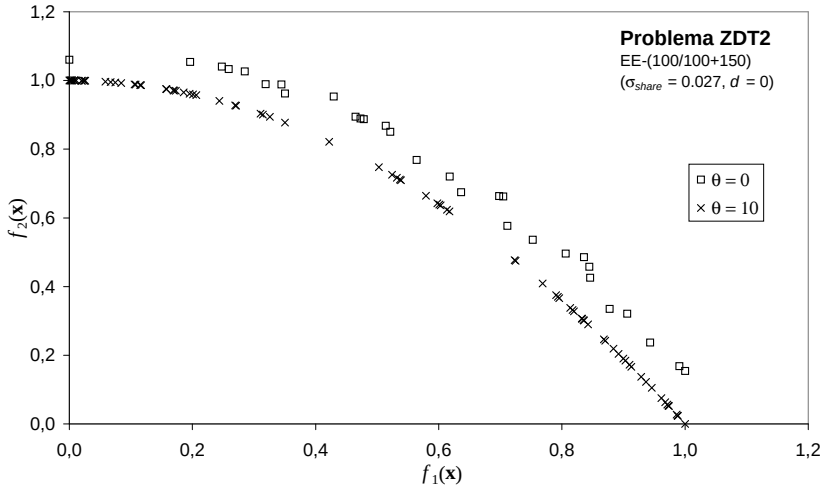

Figura 7.5: Resultados para $\theta = 0$ e $\theta = 10$ (Problema ZDT2)

problema ZDT2, com $\theta = 0$ e $\theta = 10$. É evidente que a aproximação à frente de soluções óptimas de Pareto obtida com $\theta = 10$ é claramente superior à obtida com $\theta = 0$.

7.2.5 Influência da Adaptação do Parâmetro de *Sharing* e da Selecção $+_{ps}$

O mesmo problema ZDT1 foi considerado para investigar o efeito da adaptação do parâmetro de *sharing* e da selecção no desempenho do MEES. Tal como anteriormente, uma EE-(100/100+150) com recombinação discreta nas variáveis de decisão e nos desvios padrão foi considerada. Os mesmos valores dos parâmetros foram considerados com excepção dos valores de σ_{share}. O esquema de selecção + foi também comparado com o esquema de selecção $+_{ps}$ variando a pressão de selecção ps. A Tabela 7.5 apresenta os resultados obtidos quando o valor da pressão de selecção, ps, foi variado.

É evidente que o esquema de selecção mais simples, a selecção +, é batida pelos esquemas de selecção $+_5$, $+_{10}$ e $+_{15}$. Além disso, para este problema, os melhores resultados

ZDT1	$+_2$	$+_5$	$+_{10}$	$+_{15}$	$+_{20}$	$+_{25}$	$+_{50}$	$+_{90}$	$+_{100}$
$+$	[100,3.2]	[92.5,96.1]	[93.6,97.0]	[97.0,97.6]	[98.5,95.0]	[98.1,93.1]	[99.0,91.8]	[96.7,92.6]	[99.0,87.3]
$+_2$	-	[3.2,100]	[3.2,100]	[3.2,100]	[3.2,100]	[3.2,100]	[3.2,100]	[3.2,100]	[3.2,100]
$+_5$	-	-	[94.0,93.1]	[96.1,94.0]	[98.1,88.2]	[97.4,89.6]	[99.1,85.5]	[98.1,90.8]	[99.4,77.7]
$+_{10}$	-	-	-	[96.6,95.6]	[98.0,92.0]	[97.1,89.3]	[97.9,85.5]	[97.6,91.2]	[98.9,81.2]
$+_{15}$	-	-	-	-	[97.6,91.7]	[98.5,90.9]	[98.4,89.5]	[97.3,92.5]	[99.6,83.4]
$+_{20}$	-	-	-	-	-	[96.6,96.3]	[97.5,93.2]	[95.7,95.5]	[97.5,88.5]
$+_{25}$	-	-	-	-	-	-	[97.6,92.3]	[94.8,95.2]	[98.0,90.8]
$+_{50}$	-	-	-	-	-	-	-	[94.7,98.3]	[97.2,92.8]
$+_{90}$	-	-	-	-	-	-	-	-	[98.5,89.5]

Tabela 7.5: Influência da Pressão de Selecção (Problema ZDT1)

foram obtidos com esquema de selecção $+_5$.

A Tabela 7.6 apresenta os resultados obtidos para o problema ZDT1 para $\sigma_{share} = 0.014$, $\sigma_{share} = 0.027$ e utilizando adaptação do parâmetro de *sharing* ($\alpha = 0$, $\alpha = 0.5$ e $\alpha = 1$). Nesta tabela, o esquema de selecção + foi considerado. Na Tabela 7.7 são feitas comparações com os resultados obtidos utilizando o esquema de selecção $+_5$.

As Tabelas 7.6 e 7.7 comparam os resultados obtidos para diferentes valores de σ_{share}

		+			
ZDT1		$\sigma_{sh} = 0.027$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1$
+	$\sigma_{sh} = 0.014$	[92.6,97.3]	[92.9,97.8]	[92.9,97.8]	[96.0,96.0]
	$\sigma_{sh} = 0.027$	-	[95.5,93.5]	[96.5,93.8]	[99.0,93.7]
	$\alpha = 0$	-	-	[94.3,93.4]	[94.9,93.7]
	$\alpha = 0.5$		-	-	[94.6,93.6]

Tabela 7.6: Influência da Adaptação do Parâmetro de *Sharing* (Problema ZDT1)

		$+_5$				
ZDT1		$\sigma_{sh} = 0.014$	$\sigma_{sh} = 0.027$	$\alpha = 0$	$\alpha = 0.5$	$\alpha = 1$
$+$	$\sigma_{sh} = 0.014$	[91.2,98.0]	[85.7,99.4]	[92.9,97.9]	[94.6,95.5]	[92.3,98.3]
	$\sigma_{sh} = 0.027$	[94.4,94.9]	[92.5,96.1]	[97.5,95.8]	[97.1,93.3]	[94.9,94.8]
	$\alpha = 0$	[93.9,95.1]	[91.9,96.5]	[95.2,95.5]	[95.9,94.9]	[94.0,94.8]
	$\alpha = 0.5$	[93.9,95.8]	[89.7,96.6]	[95.0,97.4]	[96.0,95.5]	[95.2,93.9]
	$\alpha = 1$	[91.4,97.0]	[88.5,98.3]	[92.0,95.3]	[96.2,97.3]	[91.6,95.2]
$+_5$	$\sigma_{sh} = 0.014$	-	[92.2,95.0]	[95.6,93.3]	[99.1,93.5]	[95.9,92.4]
	$\sigma_{sh} = 0.027$	-	-	[97.2,91.7]	[97.7,89.5]	[96.8,89.7]
	$\alpha = 0$	-	-	-	[97.4,93.6]	[94.7,94.4]
	$\alpha = 0.5$	-	-	-	-	[93.8,97.6]

Tabela 7.7: Influência da Adaptação do Parâmetro de *Sharing* e da Pressão de Selecção (Problema ZDT1)

e os obtidos utilizando o esquema de adaptação do parâmetro de *sharing*. É conhecida a dificuldade relacionada com a escolha de um valor para o parâmetro de *sharing*, o σ_{share}. Neste trabalho diversos valores de σ_{share} foram testados. Os melhores resultados foram obtidos para $\sigma_{share} = 0.027$. O esquema de adaptação do parâmetro de *sharing* evita a necessidade de encontrar um valor adequado para o σ_{share}. As comparações efectuadas mostram que, apesar dos melhores resultados obtidos para este problema terem sido obtidos para $\sigma_{share} = 0.027$, os resultados obtidos com o esquema proposto não diferem grandemente para valores de α próximos de zero. Para além disso, o esquema adaptativo é melhor quando comparado com os outros valores de σ_{share} testados. De notar que alguma experimentação pode ser necessária para encontrar os melhores valores de α. No entanto, pode ser visto, pelos resultados obtidos, que o algoritmo é mais sensível a alterações nos valores de σ_{share} do que a alterações no valores de α.

7.2.6 Comparação com outros Algoritmos Multi-objectivo

O *Multiobjective Elitist Evolution Strategy* (MEES) foi comparado com dez algoritmos para os problemas ZDT1, ZDT2, ZDT3, ZDT4, ZDT5 e ZDT6. Alguns destes resultados foram obtidos e publicados por Zitzler *et al.* [ZDT00] e Knowles e Corne [KC99]. Os algoritmos considerados foram:

- RAND: algoritmo de procura aleatória;
- MOGA: Algoritmo Genético multi-objectivo de Fonseca e Fleming;
- NPGA: *Niched Pareto Genetic Algorithm*;
- HLGA: abordagem baseada em somas pesadas de Hajela e Lin;
- VEGA: *Vector Evaluated Genetic Algorithm*;
- NSGA: *Nondominated Sorting Genetic Algorithm*;
- SOEA: AE uni-objectivo utilizando a agregação de soma pesada;
- SPEA: *Strength Pareto Evolutionary Algorithm*;
- PAES: *Pareto Archived Evolution Strategy*;
- NSGA2: *Nondominated Sorting Genetic Algorithm-II*.

Para o MEES, uma EE-(100/100+150) com recombinação discreta nas variáveis de decisão e desvios padrão foi considerada. O MEES foi aplicado com e sem elitismo (MEES_0 e MEES_{10}, respectivamente). Os parâmetros d e σ_{share} foram fixados em 0 e 0.027, respectivamente. O critério de paragem foi terminar a procura após 100 gerações.

Como está descrito com mais detalhe em [ZDT00], para os algoritmos MOGA, NPGA, HLGA, VEGA, NSGA e SPEA, o tamanho da população foi 100 (para o algoritmo SPEA

o tamanho da população foi 80 com um conjunto de soluções não dominadas externo de 20 pontos). As probabilidades de recombinação e de mutação foram de 0.8 e 0.01, respectivamente. O parâmetro de *sharing* foi fixado em 0.48862. Diversas variantes do PAES foram consideradas por Knowles e Corne [KC99] para comparar o seu algoritmo com outras abordagens. No entanto, neste estudo apenas se considera o PAES com uma população externa (o arquivo) de 98 indivíduos. Duas variantes do algoritmo são consideradas: o PAES com representação binária padrão das variáveis de decisão (PAES) e o PAES com representação em código *Gray* das variáveis de decisão (PAES_{Gray}). A razão de mutação foi fixada em 1%. O PAES utiliza um mecanismo de *crowding* para controlar a diversidade das soluções. O parâmetro de *crowding* foi fixado em $l = 5$ (definindo 1024 hipercubos). O NSGA2 foi aplicado quer com representação real das variáveis de decisão (NSGA2_r) quer com representação binária das variáveis de decisão (NSGA2_b) como foi proposto por Deb *et al.* [Deb00]. Em ambas as abordagens a probabilidade de recombinação foi de 0.9. A probabilidade de mutação para o NSGA2_r foi $1/n$ enquanto que para o NSGA2_b foi $1/b$ (onde n é o número de variáveis de decisão e b é o tamanho da sequência de dígitos binários). Para o NSGA2_r, os índices das distribuições para a recombinação e para a mutação foram iguais a 20. Para todos os algoritmos (com excepção de MEES_0 e do MEES_{10}), o número máximo de gerações foi 250. Todos os algoritmos foram executados 30 vezes para cada problema teste e, para cada execução, o conjunto de todas as soluções não dominadas geradas durante a procura foram consideradas. O número de cálculos das funções objectivo foi o mesmo para todos os algoritmos (aproximadamente, 25000 avaliações). O número de cálculos de funções objectivo requerido pelo MEES foi inferior ao dos outros algoritmos (aproximadamente, 15000 avaliações).

Duas classes de algoritmos podem ser distinguidas: aqueles que não utilizam elitismo (MOGA, NPGA, HLGA, VEGA, NSGA e MEES_0) e aqueles que utilizam, explicitamente, elitismo na procura (SPEA, PAES, PAES_{Gray}, NSGA2_r, NSGA2_b e MEES_{10}). As Tabelas

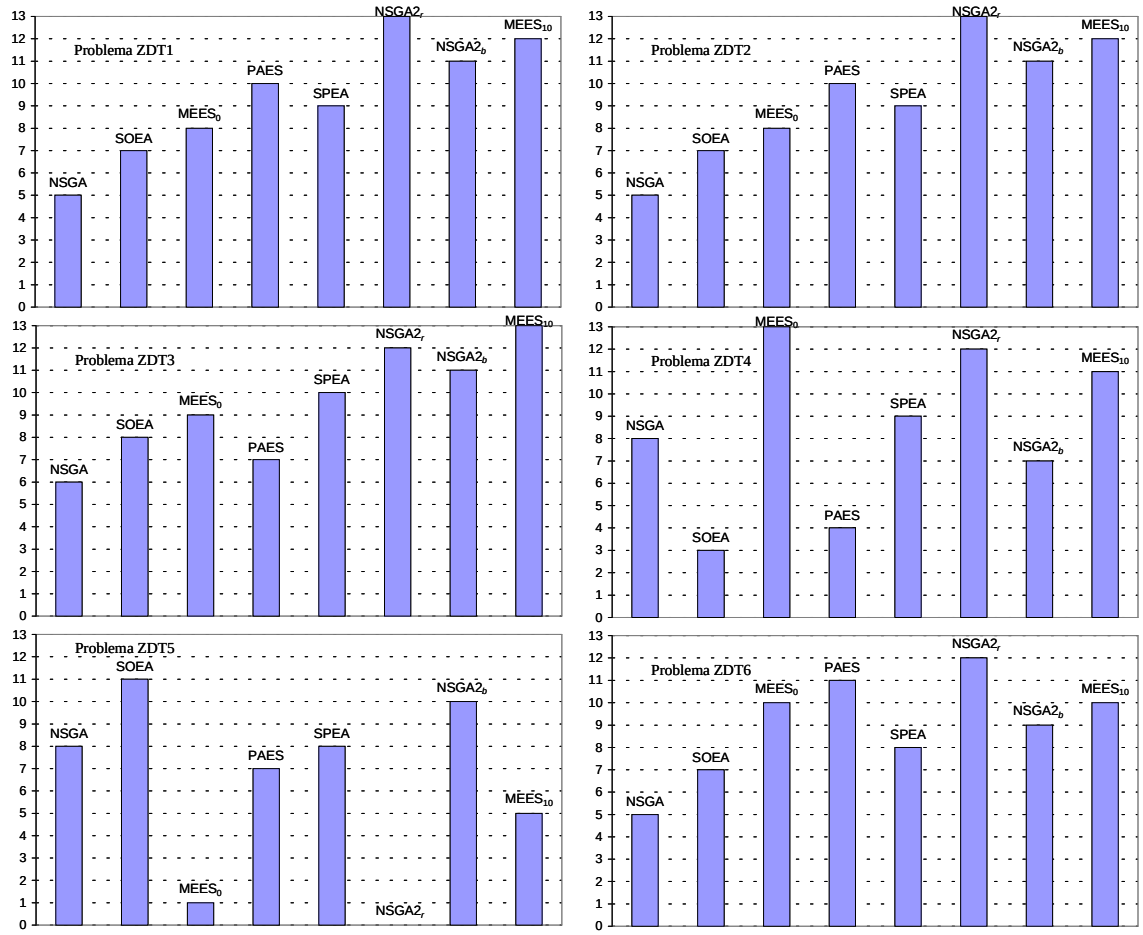

Figura 7.6: Comparação entre Algoritmos (Problemas ZDT1, ZDT2, ZDT3, ZDT4, ZDT5 e ZDT6)

7.8 a 7.18 apresentam os resultados da comparação destes algoritmos com o MEES para cada um dos problemas considerados. Para cada problema, os resultados são apresentados em duas tabelas: uma com a comparação entre os algoritmos não elitistas (Tabelas 7.8, 7.10, 7.12, 7.14, 7.16 e 7.18) e outra com a comparação de todos os algoritmos (Tabelas 7.9, 7.11, 7.13, 7.15, 7.17 e 7.19).

A Figura 7.6 apresenta, para cada problema e para cada algoritmo, o número de vezes em que um algoritmo não é "batido" pelos outros (cada algoritmo é comparado com os

ZDT1	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[100,0]	[100,9.2]	[100,10]	[100,8.3]	[100,11.4]	[100,1.3]	[75.1,37.4]
RAND	-	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	-	-	[72.2,78.7]	[21.5,93.7]	[15.2,100]	[9.6,100]	[9.3,100]
NPGA	-	-	-	[27.6,96.6]	[12.5,100]	[10.5,100]	[10.1,100]
HLGA	-	-	-	-	[75.2,77.5]	[8.8,100]	[8.4,100]
VEGA	-	-	-	-	-	[12,100]	[11.5,100]
NSGA	-	-	-	-	-	-	[1.3,100]

Tabela 7.8: Comparação entre Algoritmos Não Elitistas (Problema ZDT1)

restantes treze algoritmos). Nesta figura, por simplificação, apenas se apresentam os resultados para os algoritmos com melhor desempenho global (NSGA, SOEA, $MEES_0$, PAES, SPEA, $NSGA2_r$, $NSGA2_b$ e $MEES_{10}$).

Considerando-se apenas os resultados obtidos com algoritmos não elitistas, os melhores resultados foram obtidos pelo $MEES_0$ em todos os problemas com excepção do problema ZDT5. Para além disso, o $MEES_0$ ultrapassou todas os outros algoritmos, incluindo as abordagens elitistas, no problema ZDT4. O $MEES_0$ também exibiu um bom desempenho no problema ZDT6, ultrapassando o $MEES_{10}$. Comparando os resultados obtidos pelas duas abordagens que utilizam representação real das variáveis de decisão, é claro que o $NSGA2_r$ ultrapassou o $MEES_{10}$ nos problemas ZDT1 e ZDT2. No entanto, o contrário aconteceu com o problema ZDT3. Contudo, o $MEES_{10}$ bateu o SPEA em todos os problemas considerados. O desempenho do $MEES_{10}$ no problema ZDT5 foi claramente inferior aos outros algoritmos.

A Tabela 7.20 resume os resultados obtidos pelos diferentes algoritmos nos problemas considerados. Os valores da tabela, para cada algoritmo, são as médias da diferença, para cada comparação, entre a percentagem do espaço dos objectivos em que um algoritmo não

ZDT1	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[48.8,51.4]	[86.5,50.5]	[1.7,100]	[2.1,100]	[2.4,99.1]	[1.7,100]
RAND	[43.9,57.1]	[44.5,56.3]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	[45.8,54.8]	[42.6,54.5]	[9.2,100]	[10.1,100]	[11.9,91.5]	[9.2,100]
NPGA	[46.8,53.4]	[47.1,53.3]	[9.9,100]	[10.4,100]	[12.5,91.3]	[9.9,100]
HLGA	[47.5,53]	[48.3,52.7]	[8.3,100]	[11.7,100]	[17.1,94.8]	[8.3,100]
VEGA	[47.4,53]	[47.8,52.8]	[11.3,100]	[12.5,100]	[14.7,89.7]	[11.2,100]
NSGA	[48.4,51.8]	[49.9,51.6]	[1.3,100]	[2,100]	[2.6,99.4]	[1.3,100]
SOEA	[48.9,51.3]	[64.6,48.9]	[1,100]	[0,100]	[0,100]	[0,100]
PAES	-	[100,21.7]	[51.3,48.9]	[0,100]	[45.9,94.5]	[12.6,100]
$PAES_{Gray}$	-	-	[17.7,100]	[0,100]	[0,100]	[0,100]
SPEA	-	-	-	[2.1,100]	[2.7,98.6]	[1.9,100]
$NSGA2_r$	-	-	-	-	[100,0]	[100,28.4]
$NSGA2_b$	-	-	-	-	-	[9.7,98.5]

Tabela 7.9: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT1)

ZDT2	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[100,0]	[100,6.7]	[100,6.6]	[100,16.3]	[100,3.3]	[100,3.5]	[100,0]
RAND	-	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	-	-	[8.3,100]	[9.1,100]	[8.5,100]	[7.3,100]	[6.9,100]
NPGA	-	-	-	[61.4,75.7]	[29.3,88.3]	[8.1,100]	[7.1,100]
HLGA	-	-	-	-	[25.7,100]	[19,100]	[17.2,100]
VEGA	-	-	-	-	-	[4.6,100]	[3.7,100]
NSGA	-	-	-	-	-	-	[4.5,100]

Tabela 7.10: Comparação entre Algoritmos Não Elitistas (Problema ZDT2)

ZDT2	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[49,51.3]	[75.7,50]	[17.8,97.3]	[1.5,100]	[1.8,100]	[1.5,100]
RAND	[43.9,57.1]	[46.1,54.4]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	[45.8,54.8]	[47.4,53]	[6.8,100]	[6.8,100]	[8,100]	[6.6,100]
NPGA	[47.4,52.8]	[48.5,51.7]	[6.7,100]	[6.4,100]	[6.6,100]	[6.4,100]
HLGA	[47.6,52.7]	[48.7,51.6]	[16.5,100]	[16.1,100]	[16.3,100]	[15.9,100]
VEGA	[47.9,52.2]	[49,51.3]	[3.4,100]	[3.4,100]	[3.5,100]	[3.2,100]
NSGA	[48.4,51.7]	[49.5,50.9]	[3.6,100]	[3.6,100]	[3.7,100]	[3.4,100]
SOEA	[48.8,51.4]	[50.5,50.3]	[0,100]	[0,100]	[0,100]	[0,100]
PAES	-	[100,46.3]	[51.3,48.9]	[0,100]	[48.2,76.8]	[41.2,97.1]
$PAES_{Gray}$	-	-	[49.2,96.2]	[0,100]	[0,100]	[0,100]
SPEA	-	-	-	[1.9,100]	[2.2,100]	[1.9,100]
$NSGA2_r$	-	-	-	-	[100,0.6]	[99,2.7]
$NSGA2_b$	-	-	-	-	-	[67.1,94.9]

Tabela 7.11: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT2)

ZDT3	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[100,0]	[100,8.2]	[100,5.3]	[100,13.7]	[100,6.6]	[100,2.4]	[66.8,52.8]
RAND	-	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	-	-	[21.8,100]	[21.6,94]	[11.4,100]	[8.4,100]	[8.2,100]
NPGA	-	-	-	[26.9,91.3]	[16.4,91.9]	[5.5,100]	[5.3,100]
HLGA	-	-	-	-	[55.9,82.9]	[14.3,100]	[13.9,100]
VEGA	-	-	-	-	-	[6.9,100]	[6.7,100]
NSGA	-	-	-	-	-	-	[19.5,83.8]

Tabela 7.12: Comparação entre Algoritmos Não Elitistas (Problema ZDT3)

ZDT3	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[63.4,48.8]	[71.3,36.3]	[2.4,100]	[2.7,100]	[2.8,98.4]	[2.7,100]
RAND	[48.2,58.7]	[44.9,55.5]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	[44.3,56.1]	[46.1,54.1]	[8.2,100]	[11.5,100]	[12.5,94.2]	[11.5,100]
NPGA	[45,55.2]	[46.8,53.5]	[5.3,100]	[5.7,100]	[7.5,96.3]	[5.7,100]
HLGA	[45.8,54.6]	[47.7,52.9]	[13.7,100]	[15.6,100]	[17.6,92.2]	[15.6,100]
VEGA	[45.8,54.5]	[47.5,52.9]	[6.6,100]	[7.3,100]	[8,96.8]	[7.3,100]
NSGA	[50,52.3]	[62.6,51.9]	[2.4,100]	[2.7,100]	[3.4,98.4]	[2.7,100]
SOEA	[62.1,49.9]	[75.7,35.7]	[2.2,99.3]	[0,100]	[0,100]	[0,100]
PAES	-	[100,3]	[45.4,65.3]	[0.4,100]	[32.5,85.8]	[10.8,100]
$PAES_{Gray}$	-	-	[24.6,99.6]	[0,100]	[0,100]	[0,100]
SPEA	-	-	-	[0.9,100]	[1.1,99.5]	[0.9,100]
$NSGA2_r$	-	-	-	-	[100,0.3]	[30.6,100]
$NSGA2_b$	-	-	-	-	-	[28.4,100]

Tabela 7.13: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT3)

ZDT4	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[100,0]	[100,6]	[100,6.6]	[100,32.7]	[100,9.8]	[100,16.7]	[100,0]
RAND	-	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	-	-	[6.5,100]	[17,100]	[6.3,100]	[6.1,100]	[6.9,100]
NPGA	-	-	-	[42.8,95.8]	[7.7,100]	[6.9,100]	[22.5,100]
HLGA	-	-	-	-	[87.6,47.4]	[33.1,100]	[87.7,46.7]
VEGA	-	-	-	-	-	[10.6,100]	[94.7,18.4]
NSGA	-	-	-	-	-	-	[100,0]

Tabela 7.14: Comparação entre Algoritmos Não Elitistas (Problema ZDT4)

ZDT4	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[100,0]	[100,0]	[100,13.7]	[96.5,58.1]	[100,13.2]	[100,27.9]
RAND	[19.6,91.6]	[22,84.7]	[0,100]	[0,100]	[0,100]	[0,100]
MOGA	[13.9,90.4]	[15.3,88.6]	[6.1,100]	[6,100]	[6.1,100]	[6,100]
NPGA	[33.4,70.5]	[20.3,83.8]	[6.9,100]	[6.6,100]	[7,100]	[6.6,100]
HLGA	[100,8.4]	[45.6,55.8]	[33.1,100]	[32.7,100]	[33.3,100]	[32.7,100]
VEGA	[100,56.2]	[31.2,71.8]	[10.6,100]	[9.8,100]	[11.2,100]	[9.8,100]
NSGA	[100,0]	[42.1,60]	[82.2,100]	[16.7,100]	[100,18.2]	[16.7,100]
SOEA	[48.1,59.8]	[27.7,72.3]	[0,100]	[0,100]	[4,100]	[0,100]
PAES	-	[32,100]	[0,100]	[0,100]	[0,100]	[0,100]
$PAES_{Gray}$	-	-	[60.6,42.9]	[0,100]	[64.7,38.3]	[0,100]
SPEA	-	-	-	[13.7,100]	[100,16.2]	[13.7,100]
$NSGA2_r$	-	-	-	-	[100,13.2]	[99.1,54.7]
$NSGA2_b$	-	-	-	-	-	[13.2,100]

Tabela 7.15: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT4)

ZDT5	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[100,31.7]	[10.4,95.8]	[0,100]	[23.8,78.1]	[0,100]	[0,100]	[0,100]
RAND	-	[1.6,100]	[0,100]	[20.9,81]	[0,100]	[0,100]	[0,100]
MOGA	-	-	[82.1,72.7]	[75.8,44.1]	[68.4,41.9]	[32.5,71]	[1.8,100]
NPGA	-	-	-	[79.3,49.6]	[65.1,41.8]	[37.3,64.7]	[12.6,100]
HLGA	-	-	-	-	[57.3,46.6]	[39.1,64.3]	[0,100]
VEGA	-	-	-	-	-	[0,100]	[0,100]
NSGA	-	-	-	-	-	-	[17.8,93.1]

Tabela 7.16: Comparação entre Algoritmos Não Elitistas (Problema ZDT5)

ZDT5	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[0,100]	-	[0,100]	-	[0,100]	[0,100]
RAND	[0,100]	-	[0,100]	-	[0,100]	[0,100]
MOGA	[47.9,57.7]	-	[7.2,100]	-	[2.7,100]	[55.8,53]
NPGA	[51.5,54.6]	-	[14.5,100]	-	[12.6,100]	[54,54.2]
HLGA	[53,51.6]	-	[0,100]	-	[0,100]	[50.9,51.8]
VEGA	[67.6,99.8]	-	[0,100]	-	[0,100]	[56.4,91.6]
NSGA	[100,0]	-	[54.3,100]	-	[38.6,100]	[100,0]
SOEA	[100,0]	-	[93,36.7]	-	[78.9,45.2]	[100,0]
PAES	-	-	[100,0]	-	[0,100]	[92.9,58.6]
$PAES_{Gray}$	-	-	-	-	-	-
SPEA	-	-	-	-	[73.8,100]	[100,0]
$NSGA2_r$	-	-	-	-	-	-
$NSGA2_b$	-	-	-	-	-	[100,0]

Tabela 7.17: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT5)

ZDT6	RAND	MOGA	NPGA	HLGA	VEGA	NSGA	SOEA
$MEES_0$	[81.1,19.5]	[76.4,26]	[67.7,33.3]	[63.8,36.9]	[64.6,35.7]	[58.8,41.5]	[52.8,47.2]
RAND	-	[6.2,100]	[0,100]	[0,100]	[6.6,100]	[0,100]	[0,100]
MOGA	-	-	[0,100]	[0,100]	[17.1,100]	[0,100]	[0,100]
NPGA	-	-	-	[19,100]	[38.6,65.1]	[13.5,100]	[11.5,100]
HLGA	-	-	-	-	[100,11.2]	[4.9,100]	[4,100]
VEGA	-	-	-	-	-	[0,100]	[0,100]
NSGA	-	-	-	-	-	-	[8,100]

Tabela 7.18: Comparação entre Algoritmos Não Elitistas (Problema ZDT6)

ZDT6	PAES	$PAES_{Gray}$	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
$MEES_0$	[45.7,73.1]	[49.8,50.4]	[51.3,49.2]	[71.3,79.4]	[100,13.9]	[85.1,47]
RAND	[6.6,97.5]	[0,100]	[0,100]	[19.5,81.1]	[19.7,80.9]	[19.5,81.1]
MOGA	[12.1,100]	[0,100]	[0,100]	[26,76.4]	[26.3,76.1]	[2.6,76.4]
NPGA	[23.2,78.9]	[0,100]	[9.4,100]	[22.7,78.7]	[23.2,78.3]	[22.7,78.7]
HLGA	[26.7,80.2]	[0,100]	[3.5,100]	[25.1,76]	[25.6,75.4]	[25,76]
VEGA	[26.2,100]	[0,100]	[0,100]	[34.8,65.4]	[35.5,64.8]	[34.8,65.4]
NSGA	[32.9,67.7]	[1.1,100]	[5.2,100]	[15.5,85.3]	[16,85]	[15.5,85.3]
SOEA	[61,39]	[10.8,95.5]	[0,100]	[8.9,91.1]	[9.1,90.9]	[8.9,91.1]
PAES	-	[67.6,37.7]	[41.1,60]	[69.2,41.3]	[100,0]	[80.6,29]
$PAES_{Gray}$	-	-	[93,15.4]	[47.7,52.7]	[48.3,52]	[47.4,52.7]
SPEA	-	-	-	[17.1,85.4]	[17.1,85.4]	[17.1,85.4]
$NSGA2_r$	-	-	-	-	[100,0]	[87.4,25.7]
$NSGA2_b$	-	-	-	-	-	[0,100]

Tabela 7.19: Comparação entre Algoritmos Elitistas e Não Elitistas (Problema ZDT6)

	$MEES_0$	NSGA	SOEA	PAES	SPEA	$NSGA2_r$	$NSGA2_b$	$MEES_{10}$
ZDT1	-36.8	-84.7	-48.6	-32.2	+0.2	+95.1	+35.6	+71.4
ZDT2	-25.8	-83.0	-58.2	-24.8	-2.9	+98.4	+42.0	+54.4
ZDT3	-37.4	-78.8	-47.8	-40.9	+2.2	+74.8	+38.7	+89.1
ZDT4	+81.0	+2.0	-86.8	-84.0	+6.1	+66.1	-32.9	+48.6
ZDT5	-100.0	+19.6	+77.6	-11.0	+10.5	-	+59.0	-55.7
ZDT6	+16.2	-63.9	-34.0	+28.7	+1.0	+51.7	-23.9	+24.1

Tabela 7.20: Comparação do Desempenho Global

é ultrapassado quando comparado com os restantes algoritmos. Portanto, para cada problema, os algoritmos com melhor desempenho são aqueles que apresentam maiores valores desta medida. Logo, a partir desta tabela, é claro que, em geral, o elitismo é útil para guiar a procura. Para além disso, a representação real das variáveis de decisão permite globalmente atingir melhores resultados. O algoritmo não elitista com melhor desempenho parece ser o MEES_0 que ultrapassou todas as outras abordagens no problema ZDT4. Em todos os problemas, os resultados mais consistentes foram obtidos pelo NSGA2_r com excepção dos problemas ZDT3 e ZDT4. O MEES_{10} ultrapassou o NSGA2_r no problema ZDT3.

7.3 Discussão dos Resultados

O novo algoritmo para optimização multi-objectivo incorpora as principais características das Estratégias Evolutivas, tal como a representação real das variáveis de decisão e adaptação dos parâmetros de procura. No entanto, algumas técnicas foram introduzidas que distinguem esta abordagem, nomeadamente, um esquema adaptativo do parâmetro de *sharing*, um operador de selecção que permite controlar a pressão de selecção (ps), um parâmetro θ definindo o nível de elitismo e um parâmetro (d) que define a concentração de soluções na população secundária que aproximam o conjunto de soluções óptimas de Pareto.

No que diz respeito à adaptação do parâmetro de *sharing*, o esquema proposto evita a necessidade de encontrar o "melhor" valor para o parâmetro σ_{share}. O operador de selecção incorpora soluções de diferentes frentes, através de uma progressão geométrica, permitindo, portanto, obter uma grande diversidade enquanto é controlada a pressão de selecção. O estudo do número de soluções da população secundária que são introduzidas na procura, permite a definição adequada do nível de elitismo. O parâmetro d, enquanto define a

distância mínima desejável entre soluções no espaço dos objectivos, permite controlar o tamanho da população secundária.

O algoritmo foi testado com vários problemas teste multi-objectivo para investigar a influência de alguns factores no seu desempenho, bem como para comparar o seu desempenho com outras abordagens evolucionárias. Como esperado, os resultados indicaram que a recombinação e o elitismo são essenciais para a obtenção de boas aproximações à frente de soluções óptimas de Pareto. O algoritmo sem elitismo (MEES_0) ultrapassou as outras abordagens não elitistas (MOGA, NPGA, HLGA, VEGA e NSGA) em todos os problemas considerados. O algoritmo com elitismo (MEES_{10}) ultrapassou o NSGA2_r num único problema. No entanto, NSGA2_r teve um desempenho melhor nos problemas ZDT1, ZDT2 e ZDT6. Deve-se salientar que o número total de avaliações da função objectivo requeridas pelo MEES é inferior ao dos outros algoritmos com os quais foi comparado.

Capítulo 8

Aplicações

Neste capítulo são apresentadas diversas aplicações de Algoritmos Evolucionários a problemas de engenharia formulados como problemas multi-objectivo. Na Secção 8.1 é descrita a aplicação de um Algoritmo Genético Multi-objectivo ao problema de projecto de placas laminadas com materiais isotrópicos descrito na Secção 4.2. Em seguida, na Secção 8.2 a aplicação de um AG multi-objectivo ao projecto de placas laminadas com materiais compósitos é descrito. Estes problemas de engenharia foram formulados e resolvidos anteriormente como problemas uni-objectivo. Os resultados anteriormente obtidos são comparados com os obtidos com as respectivas formulações multi-objectivo.

8.1 Optimização de Placas Laminadas com Materiais Isotrópicos

O modelo estrutural considerado na Secção 4.2.1 pode ser formulado, tal como foi anteriormente descrito, como um problema de optimização com um único objectivo. Nesta formulação o objectivo era minimizar a complacência da estrutura sujeita a algumas restrições associadas à espessura, preço e massa da estrutura. A aplicação de um AG a este problema de optimização impõe que estas restrições sejam parte de uma função de penalização. Parece então mais natural considerar uma formulação multi-objectivo do mo-

delo estrutural com quatro funções objectivo associadas à complacência, espessura, preço e massa da estrutura [COF+02]. As experiências descritas nesta secção parecem indicar que a abordagem multi-objectivo é a mais apropriada para tratar o modelo estrutural considerado.

8.1.1 Formulação do Problema

Neste problema o objectivo é projectar uma placa laminada com materiais isotrópicos como foi formulado na Secção 4.2.1. Uma abordagem alternativa para lidar com as restrições difíceis (4.2.19-4.2.20-4.2.21) é considerar um modelo multi-objectivo com quatro funções objectivo g_i, para $i = 1, 2, 3, 4$, como foi discutido na Secção 4.2.1. Assim, o problema de optimização da placa pode ser formulado como o seguinte problema multi-objectivo de programação inteira mista,

$$
\begin{aligned}
\text{Minimizar} \quad & g_1(\mathbf{t}, \mathbf{x}) = \frac{1}{2}\mathbf{f}^T\mathbf{u} \\
& g_2(\mathbf{t}, \mathbf{x}) = \sum_{i=1}^{k} t_i \\
& g_3(\mathbf{t}, \mathbf{x}) = S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij}\ \rho_j \\
& g_4(\mathbf{t}, \mathbf{x}) = S \sum_{i=1}^{k} t_i \ \sum_{j=1}^{p} x_{ij}\ \rho_j\ p_j \\
\text{sujeito a} \quad & \mathbf{K}(\mathbf{t}, \mathbf{m})\mathbf{u} = \mathbf{f} \\
& t_i^{\min} \leq t_i \leq t_i^{\max}, \quad i = 1, \ldots, k \\
& \sum_{j=1}^{p} x_{ij} = 1, \quad i = 1, \ldots, k \\
& \sum_{i=1}^{k} x_{ij} = 1, \quad j = 1, \ldots, p \\
& x_{ij} \in \{0, 1\}, \quad i = 1, \ldots, k, \quad j = 1, \ldots, p
\end{aligned}
$$

Parâmetro	Valor
Número de experiências	1
Número máximo de gerações	250
Tamanho da população	200
Probabilidade de recombinação (em dois pontos)	0.7
Probabilidade de mutação (uniforme)	0.01
Probabilidade de recombinação (uniforme baseada na ordem)	0.7
Probabilidade de mutação (sub-lista)	0.1
σ_{share}	0.01
θ	10
Distância mínima entre soluções na PS (d)	0.01

Tabela 8.1: Parâmetros do Algoritmo Genético Multi-objectivo

onde $\mathbf{t} = (t_i) \in \Re^k$, $\mathbf{m} = (m_j) \in \Re^p$ satisfaz (4.2.3), $\mathbf{x} = (x_{ij}) \in \Re^{k \times p}$, $\mathbf{K}(\mathbf{t}, \mathbf{m})$ é uma matriz $q \times q$ dada por (4.2.7) e $\mathbf{f} \in \Re^q$.

Como foi mostrado anteriormente, este problema pode ser formulado como um problema multi-objectivo onde os objectivos a minimizar são as restrições na massa, espessura e preço total da formulação com um único objectivo.

8.1.2 Descrição do Algoritmo

O Algoritmo Genético para optimização multi-objectivo utilizado foi o descrito no Capítulo 6. Neste algoritmo é utilizado um esquema de medição do desempenho das soluções que valoriza as soluções não dominadas. Para além disso, é considerado um esquema elitista com população secundária. Nesta aplicação, considerou-se o *sharing* com medição das distâncias entre soluções quer ao nível do espaço dos objectivos quer do das variáveis de decisão. A Tabela 8.1 apresenta os parâmetros do algoritmo utilizados.

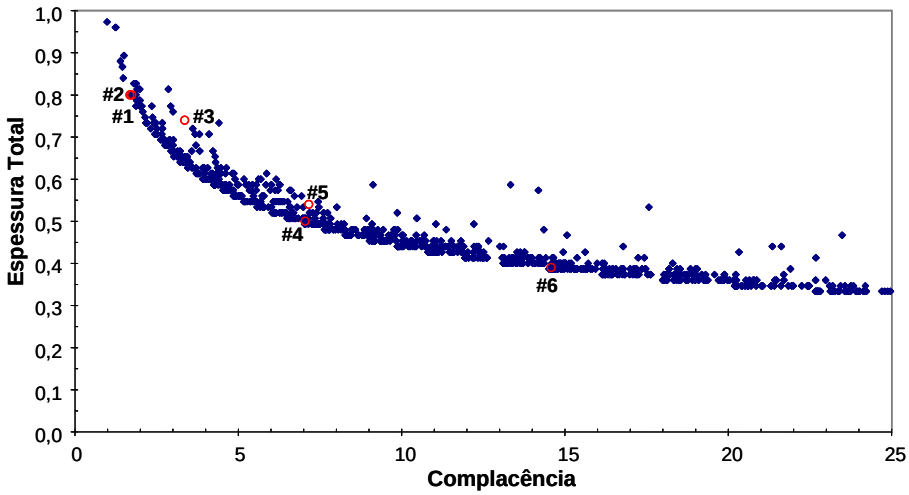

Figura 8.1: Soluções Não Dominadas (Espessura Total versus Complacência)

8.1.3 Resultados Computacionais

As Figuras 8.1, 8.2 e 8.3 mostram as soluções não dominadas presentes na PS (potenciais soluções óptimas de Pareto) ao fim de 250 gerações com *sharing* no domínio das variáveis. Os gráficos bidimensionais são apresentados como função do objectivo mais importante (a complacência). Nestas figuras, as soluções obtidas considerando a formulação do problema com um só objectivo e com restrições (Secções 4.2.1 e 4.2.3) também são indicadas (Problemas #1 a #6).

Pode-se observar que o algoritmo produz uma frente de soluções não dominadas bem definida. As soluções obtidas utilizando a formulação uni-objectivo com restrições representadas, nestes gráficos bidimensionais, pelos pontos #1 a #6 situam-se no conjunto de pontos não dominados, apesar de alguns estarem localizados em extremos do conjunto admissível de soluções. Além disso, as diversas projecções da frente permitem, com o uso de uma tabela, ao decisor escolher um ponto de operação, definindo, por exemplo, um limite na complacência que não deve ser ultrapassado e encontrando os compromissos correspondentes aos outros objectivos.

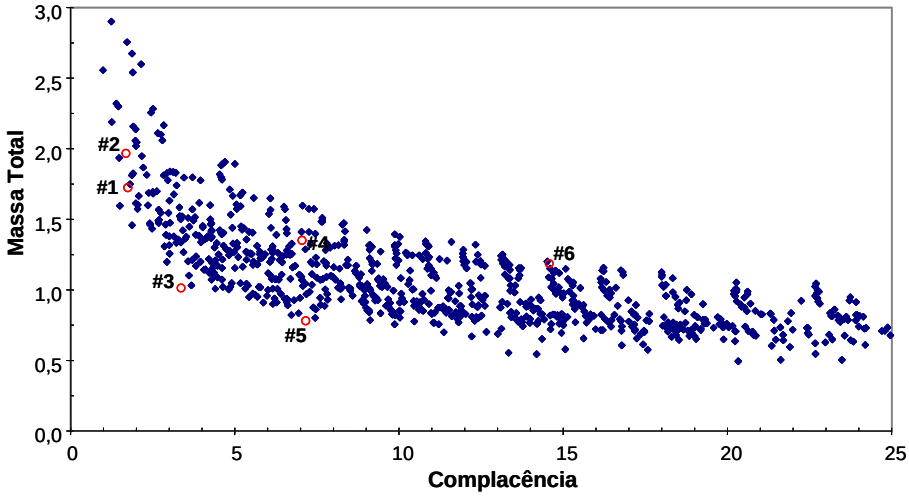

Figura 8.2: Soluções Não Dominadas (Massa Total versus Complacência)

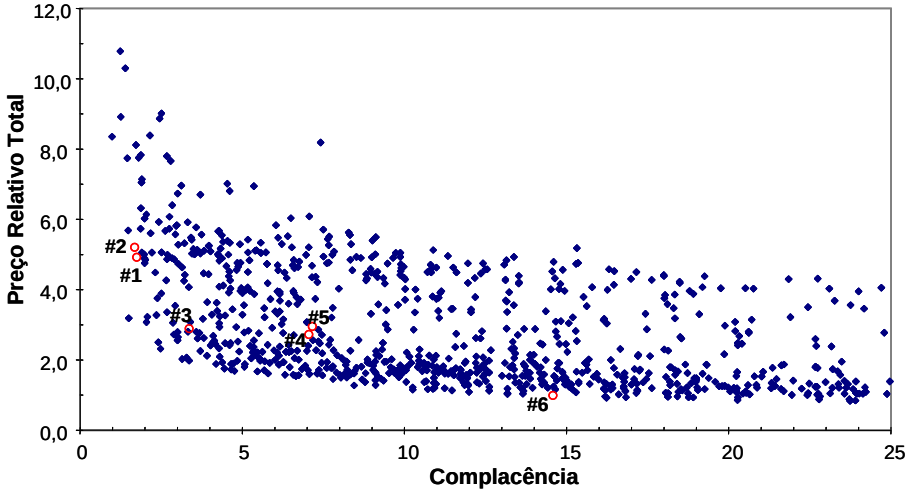

Figura 8.3: Soluções Não Dominadas (Preço Relativo Total versus Complacência)

Para avaliar a aproximação ao conjunto de soluções óptimas de Pareto, várias instâncias do problema com um objectivo foram consideradas. Nestas instâncias do problema, as restrições foram fixadas em valores extremos do conjunto admissível (ver Secção 4.2.3).

8.1.4 Discussão dos Resultados

As dificuldades na obtenção de um conjunto de soluções não dominadas que compreendessem todas as soluções geradas com a formulação com um objectivo, mostra a importância da aplicação independente de operadores genéticos às partes contínua e ordinal dos cromossomas. Para além disso, verificou-se a necessidade de manter probabilidades de mutação altas nas variáveis ordinais, sem as quais um conjunto de soluções não dominadas, que compreendesse as soluções obtidas na formulação com um objectivo, não era obtido. A utilização de pontos extremos do conjunto admissível assegura que as frentes de pontos não dominados geradas são boas aproximações ao conjunto de soluções óptimas de Pareto.

A consideração de uma população secundária constitui claramente uma importante contribuição para a boa definição das frentes de soluções não dominadas. Os resultados mostram que a implementação proposta que mantém um sub-conjunto representativo de pontos não dominados, tem vantagens óbvias uma vez que o esforço computacional é compensado pela clara definição das frentes. Deve-se salientar que todos os pontos representados nos gráficos bidimensionais são pontos não dominados. A posição relativa de alguns pontos pode parecer contradizer esta afirmação, mas estes são o resultado da projecção de uma figura a quatro dimensões em gráficos bidimensionais.

Alguns dos objectivos (preço relativo, massa e espessura) são altamente correlacionados. Surge a questão de se alguns destes objectivos podem ser considerados redundantes no sentido que se forem abandonados, o conjunto de soluções óptimas de Pareto não é alterado [GH99]. Contudo, tal requer um estudo mais aprofundado da relação entre os diferentes objectivos.

A abordagem multi-objectivo é claramente superior uma vez que numa única execução todas as soluções eficientes do problema com restrições podem ser obtidas. Para além disso, o decisor pode ver o compromisso que deve ser considerado quando escolhe um determinado ponto (solução) do conjunto de potenciais soluções óptimas de Pareto.

8.2 Optimização de Placas Laminadas com Materiais Compósitos

O modelo estrutural considerado na Secção 4.3 pode ser formulado como um problema de optimização com um único objectivo. Nesta formulação o objectivo era minimizar a complacência da estrutura sujeita a uma restrição associada ao custo da estrutura. Uma formulação multi-objectivo do modelo estrutural com duas funções objectivo associadas à complacência e ao custo da estrutura pode ser considerada [COF+02].

8.2.1 Formulação do Problema

Neste problema, o objectivo é projectar uma placa laminada com materiais compósitos como foi formulado na Secção 4.3. Na formulação multi-objectivo, a restrição de custo é considerada como um objectivo a minimizar. Assim, o problema multi-objectivo consiste em minimizar quer a complacência da placa quer o custo que lhe está associado, i.e.,

$$\text{Minimizar} \quad g_1(\mathbf{m}, \alpha) = \frac{1}{2}\mathbf{f}^T\mathbf{u} \tag{8.2.1}$$

$$g_2(\mathbf{m}, \alpha) = \sum_{j=1}^{k} m_j c(m_j) \tag{8.2.2}$$

$$\text{sujeito a} \quad \mathbf{K}(\mathbf{m}, \alpha)\mathbf{u} = \mathbf{f} \tag{8.2.3}$$

onde $\mathbf{K}(\mathbf{m}, \alpha)$ é a matriz de rigidez, $\mathbf{f}$ é o vector das forças, $\mathbf{u}$ é o vector dos deslocamentos, e g_2 é uma função que corresponde à restrição de custo.

Parâmetro	Valor
Número de experiências	1
Número máximo de gerações	100
Tamanho da População	200
Probabilidade de Recombinação	0.7
Probabilidade de Mutação	0.01
σ_{share}	0.1

Tabela 8.2: Parâmetros do Algoritmo Genético Multi-objectivo

8.2.2 Descrição do Algoritmo

Como previamente explicado, a formulação multi-objectivo não necessita que sejam consideradas restrições, uma vez que estas são incorporadas pela definição de diversos objectivos. Algoritmo Genético para optimização multi-objectivo utilizado foi o descrito no Capítulo 6. Neste algoritmo é utilizado um esquema de medição do desempenho das soluções que valoriza as soluções não dominadas. Os operador genéticos considerados foram a recombinação em dois pontos e a mutação uniforme. Na resolução do problema foi considerado *sharing* quer no domínio das variáveis de decisão quer no dos objectivos.

8.2.3 Resultados Computacionais

A lista dos materiais compósitos considerados é a apresentada anteriormente na Tabela 4.17. Nesta abordagem, depois de uma fase de experimentação dos valores dos parâmetros do algoritmo, estes foram fixados com os valores que são listados na Tabela 8.2.

As Figuras 8.4 e 8.5 mostram as soluções não dominadas obtidas ao fim de 100 gerações para o Problema #4 e Problema #5 descritos na Secção 4.3. Na obtenção dos resultados apresentados considerou-se o *sharing* ao nível das variáveis de decisão. Nestas figuras os resultados obtidos anteriormente pelo AG para a formulação com um objectivo e com

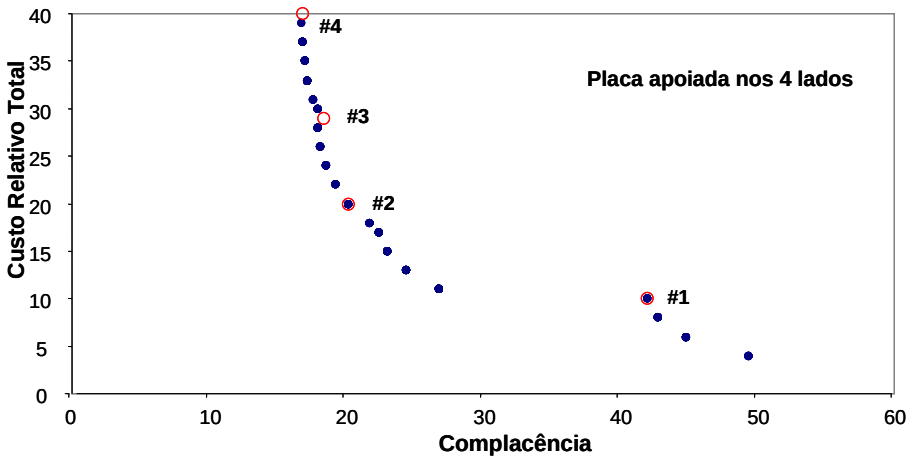

Figura 8.4: Soluções Não Dominadas (Problema #4)

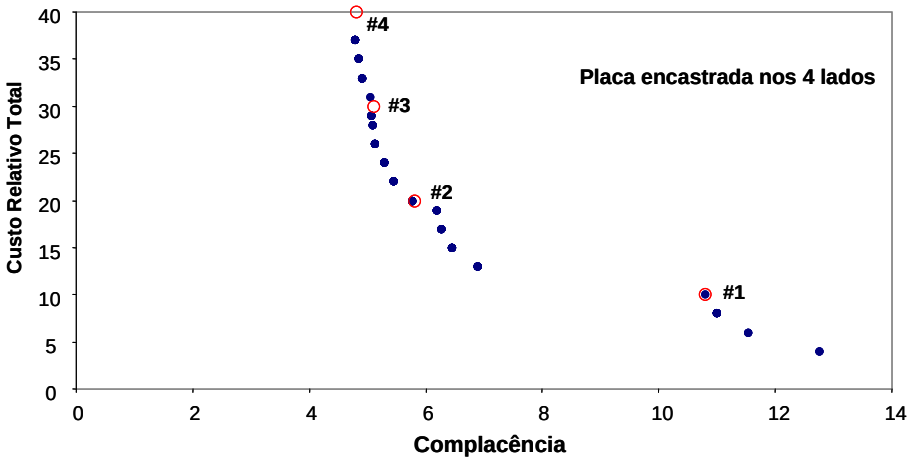

Figura 8.5: Soluções Não Dominadas (Problema #5)

restrições no custo impostas são representadas com o símbolo (o). Este resultados foram obtidos utilizando na avaliação da complacência, em ambos os casos, quatro elementos da família lagrangeana.

8.2.4 Discussão dos Resultados

Estes problemas de optimização são altamente combinatórios que, por esse motivo, implicam tempos de computação elevados, em parte devido à comunicação com o programa de elementos finitos. Para testar esta abordagem baseada em AGs, várias instâncias do problema foram formuladas, para as quais se conheciam as soluções óptimas. Logo, apesar do facto de que alguns dos materiais considerados não são utilizados na prática no projecto de laminados, as soluções obtidas têm significado físico. A aplicação de AGs permitiu a definição de gráficos bidimensionais do conjunto de potenciais soluções óptimas de Pareto (todos os pontos representados nos gráficos bidimensionais são soluções não dominadas). Verifica-se novamente a superioridade da abordagem multi-objectivo que permite ao decisor aperceber-se dos compromissos entre objectivos quando escolhe uma determinada solução.

Capítulo 9

Conclusões

O principal objectivo desta dissertação foi estudar as condições de aplicabilidade de Algoritmos Evolucionários (AEs) a problemas de optimização de engenharia. Com este propósito, duas abordagens evolucionárias, as Estratégias Evolutivas (EEs) e os Algoritmos Genéticos (AGs) foram utilizadas para resolver, numa primeira fase, problemas de optimização uni-objectivo. As características dos AEs tornam-nos particularmente adequados à resolução de problemas multi-objectivo. Portanto, numa fase posterior, diversos problemas multi-objectivo foram resolvidos utilizando abordagens evolucionárias. Os algoritmos para optimização multi-objectivo desenvolvidos incluem algumas características inovadoras. Comparações com outras abordagens evolucionárias foram realizadas.

9.1 Resultados Fundamentais em Optimização Uni-objectivo

As EEs e os AGs foram aplicados a um conjunto de problemas da área de engenharia química, o que permitiu a comparação das diferentes abordagens. Por um lado, constata-se a importância do tipo de representação das variáveis de decisão utilizada e dos mecanismos de adaptação dos parâmetros de procura; por outro lado, verifica-se a dificuldade em tratar problemas fortemente restringidos, em especial, quando existem restrições do tipo igual-

dade. Por esse motivo, diversos mecanismo de tratamento de restrições foram comparados, tendo-se verificado a superioridade do método proposto por Deb [Deb00].

Diversos problemas da área de engenharia mecânica, relacionados com o projecto de placas laminadas, foram alvo de estudo. A natureza combinatória e das variáveis de decisão destes problemas determinou a escolha da aplicação de um AG. O estudo incluiu a aplicação do AG a dois problemas distintos: a optimização de placas laminadas com materiais isotrópicos e a optimização de placas laminadas compósitas. Para cada um dos problemas, as soluções obtidas têm significado físico, validando quer o modelo utilizado quer a aplicação do AG como algoritmo de optimização. De referir que a aplicação de AGs a problemas onde existem variáveis de decisão binárias ou discretas é mais fácil do que no caso das EEs. Para além disso, a estrutura de representação das variáveis de decisão e os operadores genéticos poderão, muitas vezes, definir implicitamente restrições do problema. Subsistem, no entanto, em ambas abordagens, as dificuldades no tratamento de restrições do tipo igualdade.

Em termos globais, os AEs requerem um elevado número de avaliações da função objectivo que se traduz em elevados tempos computacionais. Este facto é particularmente relevante em problemas de engenharia em que, em geral, o cálculo do valor da função objectivo implica tempos de computação elevados, como foi o caso dos problemas considerados. Para além disso, a formulação destes problemas inclui geralmente restrições que poderão traduzir diferentes critérios. Portanto, estes inconvenientes dos AEs podem ser minimizados caso uma formulação multi-objectivo dos problemas seja considerada.

9.2 Resultados Fundamentais em Optimização Multi-objectivo

Para resolver problemas de optimização multi-objectivo foram implementados dois algoritmos incorporando algumas características inovadoras. O *Multiobjective Elitist Genetic Algorithm* (MEGA) inclui elitismo baseado numa população secundária para melhorar o seu desempenho. No MEGA é possível controlar o nível de elitismo e a concentração de soluções na população secundária, respectivamente, através dos parâmetros θ e d. O algoritmo foi testado em diversos problemas, para diversas combinações dos valores de θ e d, produzindo uma boa distribuição das soluções ao longo da frente de soluções óptimas de Pareto. O parâmetro d, que controla a concentração de soluções na aproximação ao conjunto de soluções óptimas de Pareto, permite definir compromissos entre a dimensão da população secundária e a boa distribuição das soluções no espaço dos objectivos. Por sua vez, o parâmetro θ permite regular a influência da elite na procura, o que poderá evitar situações de convergência prematura.

Uma abordagem evolucionária à optimização multi-objectivo, baseada em EEs, foi desenvolvida. No novo algoritmo, *Multiobjective Elitist Evolution Strategy* (MEES), foi feito um esforço para manter as principais características das EEs tradicionais utilizadas em optimização uni-objectivo, tais como a representação real das variáveis de decisão, a adaptação dos tamanhos do passo para mutação e a recombinação. Vários mecanismos, como o elitismo, a adaptação do parâmetro de *sharing* e um esquema de selecção geométrica provaram ser úteis na procura. A influência dos diversos parâmetros do algoritmo foi estudada para um conjunto de problemas multi-objectivo. Comparações com outros algoritmos evolucionários para optimização multi-objectivo foram realizadas, demonstrando o bom desempenho do MEES.

As formulações multi-objectivo dos problemas de optimização de placas laminadas fo-

ram resolvidas utilizando o MEGA, obtendo-se resultados superiores aos da formulação uni-objectivo.

9.3 Trabalho Futuro

As seguintes áreas serão objecto de investigação futura:

- tratamento de restrições de problemas uni-objectivo utilizando formulações multi-objectivo. Esta área tem sido alvo de investigação por parte de alguns autores como Coello Coello [CC00a, CC00b]. Embora alguma experimentação tenha sido realizada aplicando, a problemas PIMNL uni-objectivo, os algoritmos multi-objectivo propostos onde o segundo objectivo consistiu em minimizar o quadrado das violações das restrições, é necessária mais investigação;

- estudo dos parâmetros do MEES relacionados quer com a adaptação dos tamanhos do passo ao longo da procura quer com o elitismo;

- aplicação do MEES a problemas com mais de dois objectivos como os sugeridos por Deb *et al.* [DTLZ02] e comparação com outras abordagens evolucionárias, tendo em consideração, possivelmente, outras métricas;

- aplicação do MEES a problemas de engenharia e tratamento de restrições dos problemas multi-objectivo;

- esquemas que permitam a redução do número de avaliações da função objectivo efectivamente realizadas, tal como o implementado na resolução do problema de optimização de placas laminadas com materiais isotrópicos que permitiu uma redução significativa do tempo computacional requerido;

- desenvolvimento, para as aplicações informáticas realizadas, de um interface amigável com o utilizador, o que incluirá a utilização de AMPL (*A Mathematical Programming Language*) [FGK90] com o objectivo de facilitar a especificação de problemas de optimização;

- utilização de métodos estatísticos que possam ajudar, para um dado problema multi-objectivo, a identificar objectivos redundantes.

Apêndice A

Implementação dos Algoritmos Evolucionários

Para resolver problemas uni-objectivo e multi-objectivo, utilizando Algoritmos Evolucionários, foram implementadas duas aplicações informáticas: uma baseada em Estratégias Evolutivas, denominada de **EE2.0**, e outra em Algoritmos Genéticos, denominada **AG2.0**. As aplicações foram codificadas em linguagem C, funcionando em ambiente *Windows*. A versão compilada (de 32 bits) exige um sistema operativo que suporte aplicações de consola 32 bits (Windows 9X), no entanto é possível obter versões que funcionem noutros ambientes. Para isso, basta recompilar o código C que foi escrito de forma a garantir o sua fácil adaptação a outros compiladores e/ou ambientes de programação.

Todos os ficheiros utilizados e criados pelas aplicações são ficheiros de texto. Existem quatro tipos de ficheiros principais que são criados em qualquer execução:

- Iniciação - contém informação relativa à configuração utilizada;
- Resultados - contém os resultados da execução, impressos com uma determinada periodicidade;
- Estatísticas - contém estatísticas relativas à execução;
- Melhores Indivíduos - contém um determinado número dos melhores indivíduos en-

contrados.

Opcionalmente, poderão ser criados os ficheiros:

- Sharing/Multi-objectivo - contém informação relativa aos resultados obtidos quando se faz optimização multi-modal (*sharing*) ou se faz optimização multi-objectivo;
- Listagem da População - contém a listagem de todos os indivíduos da população impressa com uma determinada periodicidade;
- Teste de Esquema (apenas **AG2.0**) - contém informação relativa aos resultados obtidos de um teste de um determinado esquema;

Em seguida, neste apêndice, as aplicações são apenas descritas em termos genéricos, focando-se apenas as suas funcionalidades fundamentais.

A.1 Aplicação EE2.0

A aplicação informática **EE2.0** implementa as Estratégias Evolutivas para optimização uni-objectivo e o *Multiobjective Elitist Evolution Strategy* para optimização multi-objectivo. Os seguintes ficheiros fazem parte da aplicação:

- Global.h;
- Main.c;
- Oper_gen.h e Oper_gen.c;
- Seleccao.h e Seleccao.c;
- Recomb.h e Recomb.c;
- Mutacao.h e Mutacao.c;

- Desvios.h e Desvios.c;
- Multi_obj.h e Multi_obj.c;
- Tab_probl.h e Tab_probl.c;
- Func_obj.h e Func_obj.c;
- Calc_est.h e Calc_est.c;
- Config.h e Config.c;
- Formato.h;
- Rot_aux.h e Rot_aux.c.

Os principais detalhes e opções de implementação são descritos em seguida.

A.1.1 Tipos de Variáveis

A aplicação **EE2.0** permite a definição de três tipos distintos de variáveis o que possibilita a sua aplicação a um grande número de problemas de optimização. Os três tipos de variáveis, que podem coexistir num mesmo problema, são:

- Variáveis reais;
- Variáveis binárias;
- Variáveis inteiras.

A.1.2 Geração da População Inicial

A população inicial de progenitores é gerada de forma aleatória a partir de uma aproximação inicial ou, opcionalmente, de forma completamente aleatória. Para as variáveis

reais são gerados valores aleatórios reais, tal como para as variáveis binárias, onde cada dígito binário é escolhido aleatoriamente. Para as variáveis inteiras, cada valor é determinado aleatoriamente dentro da gama de valores definidos como válidos. A semente do gerador de números aleatórios pode ser definida.

A.1.3 Adaptação do Tamanho do Passo

É possível utilizar diversos mecanismos de adaptação, e.g., isotrópica e não isotrópica. Outros esquemas adaptativos estão também disponíveis.

A.1.4 Selecção

A selecção pode ser do tipo "+", "," ou "$+_{ps}$" e "$,_{ps}$" (do tipo geométrico).

A.1.5 Operadores

Recombinação

Recombinação discreta ou intermédia para as variáveis e/ou desvios padrão, podendo-se definir o número de indivíduos que participam na recombinação.

Mutação

A mutação baseada numa distribuição Normal para as variáveis reais e baseada numa distribuição binomial para as variáveis inteiras.

A.2 Aplicação AG2.0

A aplicação informática **AG2.0** implementa os Algoritmos Genéticos para optimização uni-objectivo e o*Multiobjective Elitist Genetic Algorithm* para optimização multi-objectivo. Os

seguintes ficheiros fazem parte da aplicação:

- Global.h;
- Main.c;
- Oper_gen.h e Oper_gen.c;
- Seleccao.h e Seleccao.c;
- Crossover.h e Crossover.c;
- Mutacao.h e Mutacao.c;
- Sharing.h e Sharing.c;
- Multi_obj.h e Multi_obj.c;
- Tab_probl.h e Tab_probl.c;
- Func_obj.h e Func_obj.c;
- Tab_mat.h;
- Conv_cod.h e Conv_cod.c;
- Calc_est.h e Calc_est.c;
- Tab_lookup.h e Tab_lookup.c;
- Config.h e Config.c;
- Formato.h;
- Rot_aux.h e Rot_aux.c.

Os principais detalhes e opções de implementação são descritos em seguida.

A.2.1 Tipos de Variáveis

A aplicação **AG2.0** permite a definição de dois tipos distintos de variáveis, o que possibilita a sua aplicação a um grande número de problemas de optimização. Os dois tipos de variáveis, que podem coexistir num mesmo problema, são:

- Variáveis reais onde a representação pode ser feita utilizando sequências de dígitos binários (código binário padrão ou código *Gray*);
- Variáveis ordinais.

As variáveis reais podem, opcionalmente, ser representadas por vectores de valores reais em vez da tradicional representação através de sequências de dígitos binários (neste caso, utilizam-se operadores genéticos apropriados a este tipo de representação, e.g., recombinação aritmética). É possível obter estatísticas relativas a um determinado esquema (apenas quando se usa exclusivamente variáveis reais e codificadas utilizando sequências de dígitos binários).

A.2.2 Geração da População Inicial

A população inicial de cromossomas é gerada de forma aleatória; i.e., para as variáveis reais com codificação binária, cada dígito binário de cada cromossoma da população é escolhido aleatoriamente, usando um gerador de números aleatórios; para as variáveis ordinais, cada valor é determinado aleatoriamente dentro da gama de valores definidos como válidos. Se as variáveis reais são representadas por vectores reais, então são gerados valores aleatórios reais. O mesmo se passa para as variáveis ordinais onde os valores são gerados aleatoriamente dentro da gama de valores possíveis para as variáveis. A semente do gerador de números aleatórios pode ser definida.

A.2.3 Medição do Desempenho

A aplicação permite optar entre a utilização de medição do desempenho proporcional com janela de escala e medição do desempenho baseada em graduações. Consoante os casos, é possível definir o tamanho da janela de escala ou a pressão de selecção.

A.2.4 Selecção

A selecção é feita por amostragem universal estocástica ou por selecção estocástica dos restantes (sem reposição).

A.2.5 Operadores Genéticos

Recombinação

O cruzamento é duplo; i.e., é múltiplo com 2 pontos de cruzamento. No caso de variáveis ordinais utiliza-se o cruzamento uniforme baseado na ordem. É possível definir a probabilidade de cruzamento.

Mutação

A mutação é feita uniformemente. No caso de variáveis ordinais utiliza-se mutação por inversão de sub-lista. É possível definir a probabilidade de mutação.

Inversão

É permitido, por opção, utilizar o operador de inversão para as variáveis reais. É possível definir a probabilidade de inversão.

A.2.6 Re-inserção

A substituição é uniforme, sendo possível escolher a proporção de indivíduos a substituir de geração em geração.

Apêndice B

Cálculo de Números Aleatórios Normalmente Distribuídos

A determinação de dois números aleatórios independentes normalmente distribuídos com média nula e variância unitária pode ser feita a partir de quaisquer dois números aleatórios gerados uniformemente no intervalo $[0, 1]$, utilizando as regras de transformação de Box e Muller [BM58]. Estas regras de transformação podem ser escritas na seguinte forma:

$$x_1 = \sqrt{-2\ln(y_1)}\sin(2\pi y_2)$$

e

$$x_2 = \sqrt{-2\ln(y_1)}\cos(2\pi y_2)$$

onde y_1 e y_2 são os dois números aleatórios gerados uniformemente no intervalo $[0, 1]$ e, x_1 e x_2 são os dois números aleatórios normalmente distribuídos com média nula e variância unitária. Para se obter dois números aleatórios z_1 e z_2 normalmente distribuídos com média nula e variância σ^2, basta multiplicar x_1 e x_2 pelo desvio padrão σ:

$$z_1 = \sigma x_1$$

e

$$z_2 = \sigma x_2$$

Apêndice C

Esquemas de Representação Binária

C.1 Código Binário Padrão

Na codificação de valores no alfabeto binário ($B = \{0, 1\}$) com l dígitos binários podem ser representados 2^l valores distintos. A representação em binário, $\langle b_1 \dots b_l \rangle_2$ com $b_i \in B$, de um valor decimal inteiro positivo $x \in \left\{0, \dots, 2^l - 1\right\}$ é determinada através do algoritmo padrão de codificação em binário. A função que descreve a codificação de um valor decimal inteiro positivo x na sua correspondente representação em binário $\langle b_1 \dots b_l \rangle_2$ pode ser escrita na seguinte forma:

$$bin : \left\{0, \dots, 2^l - 1\right\} \rightarrow B^l$$
$$bin(x) = \langle b_1 \dots b_l \rangle_2 \text{ em que } b_i = \left(x/2^{l-1}\right) //2 \text{ com } 1 \leq i \leq l$$

onde / e // representam, respectivamente, o quociente e o resto da divisão inteira.

Exemplo - pretendendo-se codificar o valor decimal inteiro 13 em binário com $l = 4$, tem-se que $bin(13) = \langle 1101 \rangle_2$, uma vez que:

$$b_1 = (13/2^3)//2 = 1//2 = 1$$
$$b_2 = (13/2^2)//2 = 3//2 = 1$$
$$b_3 = (13/2^1)//2 = 6//2 = 0$$
$$b_4 = (13/2^0)//2 = 13//2 = 1$$

A função para descodificar uma representação em binário para o correspondente valor

decimal positivo segue-se:

$$dec : B^l \rightarrow \{0, \ldots, 2^l - 1\}$$
$$dec(\langle b_1 \ldots b_l \rangle_2) = \sum_{i=0}^{l-1} b_{l-1} 2^i$$

Exemplo - sendo $l = 4$, tem-se que $dec(\langle 0011 \rangle_2) = 3$ e $dec(\langle 1111 \rangle_2) = 15$.

A função que descreve a codificação duma variável x, tal que $\underline{x} \leq x \leq \overline{x}$ ($\underline{x}$ e $\overline{x}$ representam, respectivamente, os limites inferior e superior da variável x), numa representação em binário $\langle b_1 \ldots b_l \rangle_2$ pode ser escrita na seguinte forma:

$$bin' : [\underline{x}, \overline{x}] \rightarrow B^l$$
$$bin'(x) = \langle b_1 \ldots b_l \rangle_2 \text{ em que } b_i = \left(\left(\frac{x - \underline{x}}{\overline{x} - \underline{x}} (2^l - 1) \right) / 2^{l-1} \right) // 2 \text{ com } 1 \leq i \leq l$$

onde / e // representam, respectivamente, o quociente e o resto da divisão inteira.

Exemplo - considerando a variável $1 \leq x \leq 3$, sendo $l = 4$, e pretendendo-se codificar o valor decimal 1.8 em binário, tem-se que $bin'(1.8) = \langle 0110 \rangle_2$, uma vez que:

$$b_1 = \left(\left(\frac{1.8-1}{3-1} (2^4 - 1) \right) / 2^3 \right) // 2 = (6/2^3) // 2 = 0 // 2 = 0$$
$$b_2 = \left(\left(\frac{1.8-1}{3-1} (2^4 - 1) \right) / 2^2 \right) // 2 = (6/2^2) // 2 = 1 // 2 = 1$$
$$b_3 = \left(\left(\frac{1.8-1}{3-1} (2^4 - 1) \right) / 2^1 \right) // 2 = (6/2^1) // 2 = 3 // 2 = 1$$
$$b_4 = \left(\left(\frac{1.8-1}{3-1} (2^4 - 1) \right) / 2^0 \right) // 2 = (6/2^0) // 2 = 6 // 2 = 0$$

A função para descodificar uma representação em binário para a correspondente variável segue-se:

$$dec' : B^l \rightarrow [\underline{x}, \overline{x}]$$
$$dec'(\langle b_1 \ldots b_l \rangle_2) = \underline{x} + \frac{\overline{x} - \underline{x}}{2^l - 1} \sum_{i=0}^{l-1} b_{l-1} 2^i$$

Exemplo - considerando a variável $1 \leq x \leq 3$, e sendo $l = 4$, tem-se que $dec'(\langle 1001 \rangle_2) = 2.2$ e $dec(\langle 1111 \rangle_2) = 3$.

C.2 Código Gray

A função que descreve a conversão de um valor codificado em binário padrão, $\langle b_1 \ldots b_l \rangle_2$ com $b_i \in B$, na respectiva representação em código *Gray*, $\langle g_1 \ldots g_l \rangle_2$ com $g_i \in B$, pode ser escrita na seguinte forma:

$$gray : B^l \rightarrow B^l$$
$$gray(\langle b_1 \ldots b_l \rangle_2) = \langle g_1 \ldots g_l \rangle_2 \text{ em que } g_i = \begin{cases} b_i \leftarrow i = 1 \\ b_{i-1} \oplus b_i \leftarrow 2 \leq i \leq l \end{cases}$$

Decimal	Binário	*Gray*	Decimal	Binário	*Gray*
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

Tabela C.1: Código Binário Padrão e Código *Gray*

onde $\oplus$ representa o operador booleano "ou exclusivo"[1].

Exemplo - sendo $l = 4$ e pretendendo-se converter o valor codificado em código binário padrão $\langle 1101 \rangle_2$ na sua representação em código *Gray*, tem-se que $gray(\langle 1101 \rangle_2) = \langle 1011 \rangle_2$, uma vez que:

$$\begin{aligned} g_1 &= b_1 = 1 \\ g_2 &= b_1 \oplus b_2 = 1 \oplus 1 = 0 \\ g_3 &= b_2 \oplus b_3 = 1 \oplus 0 = 1 \\ g_4 &= b_3 \oplus b_4 = 0 \oplus 1 = 1 \end{aligned}$$

A função para descodificar uma representação em código *Gray* para a correspondente representação binária padrão segue-se:

$$bin'' : B^l \rightarrow B^l$$
$$bin''(\langle g_1 \dots g_l \rangle_2) = \langle b_1 \dots b_l \rangle_2 \text{ em que } b_i = \begin{cases} g_i \leftarrow i = 1 \\ b_{i-1} \oplus g_i \leftarrow 2 \leq i \leq l \end{cases}$$

Exemplo - sendo $l = 4$, tem-se que $bin''(\langle 0011 \rangle_2) = \langle 0010 \rangle_2$ e $bin''(\langle 1111 \rangle_2) = \langle 1010 \rangle_2$.

Note-se que é também possível converter código binário padrão para o correspondente

[1] Ou exclusivo: $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$ e $1 \oplus 1 = 0$.

código *Gray* e vice-versa usando as seguintes relações e matrizes:

$$g = A_{l\times l}b \text{ e } b = A_{l\times l}^{-1}g \text{ sendo } g = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_l \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_l \end{bmatrix}, \tag{C.2.1}$$

$$A_{l\times l} = \begin{bmatrix} 1 & 0 & 0 & & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & & 0 & 0 \\ & \vdots & & \ddots & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \text{ e } A_{l\times l}^{-1} = \begin{bmatrix} 1 & 0 & 0 & & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 1 & & 0 & 0 \\ & \vdots & & \ddots & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

onde as operações de adição e multiplicação devem ser feitas em "módulo 2"[2]. A Tabela C.1 lista as representações em binário padrão e correspondentes representações em código *Gray* para $l = 4$.

[2]Adição "módulo 2": $0+0=0$, $0+1=1$, $1+0=1$ e $1+1=0$; multiplicação "módulo 2": $0\times 0=0$, $0\times 1=0$, $1\times 0=0$ e $1\times 1=1$.

Bibliografia

[AC01] M.J. Alves e J. Clímaco, *Encyclopedia of optimization*, ch. Multi-objective Mixed Integer Programming, p. 465, 2001.

[Bäc96] T. Bäck, *Evolutionary algorithms in theory and practice*, Oxford University Press, Inc., NY, 1996.

[Bak85] J.E. Baker, *Adaptive selection methods for genetic algorithms*, Proceedings of an International Conference on Genetic Algorithms and their Application (Hillsdale, New Jersey, USA), Lawrence Erlbaum Associates, 1985, pp. 101–111.

[Bak87] ———, *Reducing bias and inefficiency in the selection algorithm*, Proceedings of the Second International Conference on Genetic Algorithms and their Applications (Hillsdale, New Jersey, USA), Lawrence Erlbaum Associates, 1987, pp. 14–21.

[Ben95] M.P. Bendsøe, *Optimization of structural topology, shape, and material*, Springer, Berlin, Germany, 1995.

[Bet81] A.D. Bethke, *Genetic algorithms as function optimizers*, Ph.D. thesis, Department of Computer and Communication Sciences, University of Michigan, 1981.

[BH91] T. Bäck e F. Hoffmeister, *Extended selection mechanisms in genetic algorithms*, Proceedings of the Fourth International Conference on Genetic Algo-

rithms (San Mateo, California, USA), Morgan Kaufmann Publishers, 1991, pp. 92–99.

[BM58] G.E.P. Box e M.E. Muller, *A note on the generation of random normal deviates*, Ann. Math. Stat. **29** (1958), 610–611.

[Boo82] L.B. Booker, *Intelligent behavior as an adaptation to the task environment*, Ph.D. thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, Michigan, USA, 1982.

[Boo87] L. Booker, *Improving search in genetic algorithms*, Genetic Algorithms and Simulated Annealing (USA) (L. Davis, ed.), Morgan Kaufmann Publishers, 1987, pp. 61–73.

[Box65] M.J. Box, *A new method of constrained optimization and a comparison with other methods*, Comp. J. **8** (1965), 42–52.

[Bri81] A. Brindle, *Genetic algorithms for function optimization*, Ph.D. thesis, Computer Science Department, University of Alberta, 1981.

[BS93] M.P. Bendsøe e C.M. Soares, *Topology design of structures*, Kluwer Academic Press, Dordrecht, The Netherlands, 1993.

[BS96] J.R. Banga e W.D. Seider, *Global optimization of chemical processes using stochastic algorithms*, State of the Art in Global Optimization: Computational Methods and Applications (Dordrecht, The Netherlands) (C.A. Floudas e P. M. Pardalos, eds.), Kluwer Academic Publishers, 1996, pp. 563–583.

[Car98] M.M.F.C. Cardoso, *À procura do óptimo global*, Ph.D. thesis, Engineering Faculty, University of OPorto, Portugal, 1998.

[CC00a] C.A. Coello Coello, *Constraint-handling using an evolutionary multiobjective optimization technique*, Civil Engineering Systems **17** (2000).

[CC00b] ———, *Use of a self-adaptive penalty approach for engineering optimization problems*, Computers and Industry **41** (2000), no. 2.

[CK70] J.F. Crow e M. Kimura, *An introduction to population genetics theory*, Harper and Row, New York, 1970.

[CO98] L. Costa e P. Oliveira, *Evolutionary algorithms in optimization problems*, Tech. Report 1/98, Universidade do Minho, 1998.

[CO99] ———, *Genetic algorithms in global optimization of mixed integer non-linear problems*, Proceedings of the Genetic and Evolutionary Computation Conference GECCO99 (USA) (W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Hanavar, M. Jakiela, e R. E. Smith, eds.), Morgan Kaufmann Publishers, 1999, pp. 1773, vol. 2.

[CO01] ———, *Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems*, Computers chem. Engng. **25** (2001), 257–266.

[CO02a] ———, *An elitist genetic algorithm for multiobjective optimization*, Conference Book of MIC 2001 - Proceedings of the 4th Metaheuristics International Conference (to appear) (J.P. Sousa e M.G.C. Resende, eds.), Kluwer, 2002.

[CO02b] ———, *An evolution strategy for multiobjective optimization*, Proceedings of the Congress on Evolutionary Computation (CD edition) (Honolulu, USA), 2002.

[COF+02] L. Costa, P. Oliveira, I. Figueiredo, L. Roseiro, e R. Leal, *Multiobjective laminated plate optimization*, Evolutionary Methods for Design, Optimization and Control (Barcelona, Spain) (J. Périaux K.D. Papailiou T. Fogarty K.C. Giannakoglou, D.T. Tsahalis, ed.), CIMNE, 2002, pp. 391–396.

[Coh78] J.L. Cohon, *Multiobjective programming and planning*, Academic Press, New York, 1978.

[COI+00] L.A. Costa, P.N. Oliveira, Figueiredo I.N., L.F. Roseiro, e R.P. Leal, *Structural optimization of laminated plates with genetic algorithms*, Proceedings of

the Genetic and Evolutionary Computation Conference GECCO2000 (EUA) (D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, e H.-G. Beyer, eds.), Morgan Kaufmann Publishers, 2000, pp. 621–627.

[COIL99] L.A. Costa, P.N. Oliveira, Figueiredo I.N., e R.P. Leal, *Compliance minimization of a composite laminated plate by genetic algorithms*, Proceedings of the European Conference on Computational Mechanics ECCM99 (Munique, Alemanha), 1999, pp. 740–741.

[CSFdA96a] M.F. Cardoso, R.L. Salcedo, e S. Feyo de Azevedo, *Nonequilibrium simulated annealing: A faster approach to combinatorial minimization*, Ind. Eng. Chem. Res. (1996), no. 33, 1908–1918.

[CSFdA96b] ———, *The simplex-simulated annealing approach to continuous non-linear optimization*, Computers chem. Engng. **20** (1996), no. 9, 1065–1080.

[CSFdAB97] M.F. Cardoso, R.L. Salcedo, S. Feyo de Azevedo, e D. Barbosa, *A simulated annealing approach to the solution of minlp problems*, Computers chem. Engng. **21** (1997), no. 12, 1349–1364.

[CW92] K.J. Callahan e G.E. Weeks, *Optimum design of composite laminates using genetic algorithms*, Composites Engineering (1992), no. 2, 149–160.

[DA95] K. Deb e R.B. Agrawal, *Simulated binary crossover for continuous search space*, Complex Systems **9** (1995), no. 2, 115–148.

[DAPM00] K. Deb, S. Agrawal, A. Pratap, e T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: Nsga-ii*, Tech. Report 200001, Indian Institute of Technology, Kanpur Genetic Algorithms Laboratory (KanGAL), 2000.

[Dav91] L. Davis, *Handbook of genetic algorithms*, ch. Order-Based Genetic Algorithms and the Graph Coloring Problem, p. 72, 1991.

[Deb99] K. Deb, *Multi-objective genetic algorithms: Problem difficulties and construction of test problems*, Evolutionary Computation Journal **7** (1999), no. 3, 205–230.

[Deb00] ———, *An efficient constraint handling method for genetic algorithms*, Comp. Meth. Appl. Mech. Eng. **186** (2000), no. 2-4, 311–338.

[Deb01] ———, *Multi-objective optimization using evolutionary algorithms*, John Wiley and Sons, Ltd, Chichester, 2001.

[DG89] K. Deb e D. Goldberg, *An investigation of niche and species formation in genetic function optimization*, Proceedings of the Third International Conference on Genetic Algorithms (USA), 1989, pp. 42–50.

[DG01] K. Deb e T. Goel, *Controlled elitist non-dominated sorting genetic algorithms for better convergence*, Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, 2001, pp. 67–81.

[DGR92] U.M. Diwekar, I.E. Grossmann, e E.S. Rubin, *An minlp process synthesizer for a sequential modular simulator*, Ind. Eng. Chem. Res (1992), no. 31, 313–322.

[DJ75] K.A. De Jong, *An analysis of the behavior of a class of genetic adaptive systems*, Dissertation Abstracts International, 5140B. (University Microfilms n. 76-9381) **36** (1975), no. 10.

[DJ80] ———, *Adaptive system design: a genetic aproach*, IEEE Trans. Syst., Man. And Cyber. **10** (1980), no. 9, 566–574.

[DK95] K. Deb e A. Kumar, *Real-coded genetic algorithms with simulated binary crossover: studies on multi-modal and multi-objective problems*, Complex Systems **9** (1995), no. 6, 431–454.

[DMC96] M. Dorigo, V. Maniezzo, e A. Colomi, *The ant system: Optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics: Part B **26** (1996), no. 1, 29–41.

[DR93] U.M. Diwekar e E.S. Rubin, *Efficient handling of the implicit constraints problem for the aspen minlp synthesizer*, Ind. Eng. Chem. Res (1993), no. 32, 2006–2011.

[DTLZ02] K. Deb, L. Thiele, M. Laumanns, e E. Zitzler, *Scalable multi-objective optimization test problems*, Proceedings of the 2002 IEEE World Congress on Computational Intelligence - Congress on Evolutionary Computation (USA), 2002.

[FAC89] C.A. Floudas, A. Aggarwal, e A.R. Ciric, *Global optimum search for nonconvex nlp and minlp problems*, Computers chem. Engng. **13** (1989), no. 10, 1117–1132.

[FF93] C.M. Fonseca e P.J. Fleming, *Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization*, Proceedings of the Fifth International Conference on Genetic Algorithms (S. Forrest, ed.), Ed. San Mateo, CA: Morgan Kauffman, 1993, pp. 416–423.

[FF95] ———, *On the performance assessment and comparison of stochastic multiobjective optimizers*, Parallel Problem Solving from Nature IV (I. Rechenberg H.-P. Schwefel H.-M. Voigt, W. Ebeling, ed.), Eds Berlin: Germany: Springer, 1995, pp. 584–593.

[FF98] ———, *Multiobjective optimization and multiple constraint handling with evolutionary algorithms-part ii: Application example*, IEEE Transactions on Systems, Man, and Cybernetics: Part A: Systems and Humans **28** (1998), no. 1, 38–47.

[FFJ+98] L. Fernandes, I. Figueiredo, J. Júdice, L. Costa, e P. Oliveira, *Application of genetic algorithms to plate optimization*, Computational Mechanics, New

Trends and Applications (Barcelona, Spain) (E. Dvorkin S. R. Idelsohn, E. Oñate, ed.), CIMNE, 1998.

[FGK90] R. Fourer, D.M. Gay, e B.W. Kernighan, *A modeling language for mathematical programming*, Management Science (1990), no. 36, 519–554.

[Flo95] C.A. Floudas, *Nonlinear and mixed-integer optimization*, Oxford University Press, New York, 1995.

[Fon95] C.M.M. Fonseca, *Multiobjective genetic algorithms with application to control engineering problems*, Ph.D. thesis, University of Sheffield, 1995.

[For85] S. Forrest, *Documentation for prisoners dilemma and norms programs that use the genetic algorithm*, Tech. report, Unpublished manuscript, Ann Harbor, University of Michigan, 1985.

[FOW66] L.J. Fogel, A.J. Owens, e M.J. Walsh, *Artificial intelligence through simulated evolution*, John Wiley, New York, 1966.

[FP87] C.A. Floudas e P.M. Pardalos, *A collection of test problems for constrained global optimization algorithms*, Lecture Notes in Computer Science **455** (1987).

[GD91] D.E. Goldberg e K. Deb, *A comparative analysis of selection schemes used in genetic algorithms*, Foundations of Genetic Algorithms (USA) (G. Rawlins, ed.), Morgan Kaufmann Publishers, 1991, pp. 69–93.

[GH99] T. Gal e T. Hanne, *Consequences of dropping non essential objectives for the application of mcdm methods*, E Jor **119** (1999), 373–370.

[Gil85] A.M. Gillies, *Machine learning procedures for generating image domain feature detectors*, Tech. report, Unpublished doctoral dissertation, Ann Harbor, University of Michigan, 1985.

[GJ79] M.R. Garey e D.S. Johnson, *Computers and intractability - a guide to the theory of np-completeness*, W.H. Freeman and Company, San Francisco, 1979.

[Gol89] D.E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley, Reading, Massachusetts, 1989.

[GR87] D.E. Goldberg e J. Richardson, *Genetic algorithms with sharing for multimodal function optimization*, Proceedings of the Second International Conference on Genetic Algorithms (Hillsdale, NJ) (J.J. Grefenstette, ed.), Lawrence Erlbaum, 1987, pp. 41–49.

[Gre84] J.J. Grefenstette, *Genesis: A system for using genetic search procedures*, Proceedings of the 1984 Conference on Intelligent Systems and machines, 1984, pp. 161–165.

[GS79] I.E. Grossmann e R.W.H. Sargent, *Optimal design of multipurpose chemical plants*, Ind. Eng. Chem. Process Des. Dev. **18** (1979), no. 343.

[Haf98] R.T. Haftka, *Genetic algorithms for optimization of composite laminates*, Mechanics of Composite Materials and Structures, 2 (Troia, Portugal) (C. A. Mota Soares, C. M. Mota Soares, e M. J. M. Freitas, eds.), NATO ASI, 1998, pp. 141–152.

[Haj90] Hajela, *Genetic search - an approach to the nonconvex optimization problem*, AIAA Journal (1990), no. 28, 1205–1210.

[HN98] J. Haslinger e P. Neittaanmäki, *Finite element approximation for optimal shape design, theory and applications*, John Wiley and Sons, Chicester, England, 1998.

[HNG94] J. Horn, N. Nafploitis, e D. Goldberg, *A niched pareto genetic algorithm for multi-objective optimization*, Proceedings of the First IEEE Conference on Evolutionary Computation, 1994, pp. 82–87.

[Hol75] J.H. Holland, *Adaptation in natural and artificial systems*, Ann Arbor, The University of Michigan Press, Michigan, 1975.

[HS81] W. Hock e K. Schittkowski, *Test examples for nonlinear programming codes*, Lecture Notes in Economics and Mathematical Systems **187** (1981).

[KC99] J.D. Knowles e D.W. Corne, *Local search, multiobjective optimization and the pareto archived evolution strategy*, Proceedings of the Third Australia-Japan Joint Workshop on Intelligent and Evolutionay Systems, 1999, pp. 209–216.

[KC00] ———, *Approximating the nondominated front using the pareto archived evolution strategy*, Evolutionary Computation **8** (2000), no. 2, 149–172.

[KE95] J. Kennedy e R.C. Eberhart, *Particle swarm optimization*, Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, 1995, pp. 1942–1948, Vol. IV.

[KG87] G.R. Kocis e I.E. Grossmann, *Relaxation strategy for the structural optimization of process flow sheets*, Ind. Eng. Chem. Res. (1987), no. 26, 1869–1880.

[KG88] ———, *Global optimization of nonconvex mixed-integer nonlinear programming (minlp) problems in process synthesis*, Ind. Eng. Chem. Res. (1988), no. 27, 1407–1421.

[KG89] ———, *A modelling and decomposition strategy for the minlp optimization of process flowsheets*, Computers chem. Engng. **13** (1989), no. 7, 797–819.

[Koz92] J.R. Koza, *Genetic programming: On the programming of computers by means of natural selection*, MIT Press, Cambridge, Mass, 1992.

[Kur90] F. Kursawe, *A variant of evolution strategies for vector optimization*, Parallel Problem Solving from Nature (H.-P. Schwefel e R. Manner, eds.), Eds Berlin: Germany: Springer, 1990, pp. 193–197.

[LCOF00] R.P. Leal, L.A. Costa, P.N. Oliveira, e I.N. Figueiredo, *Algoritmos genéticos na optimização de laminados*, Actas do VI Congresso Nacional de Mecânica Aplicada e Computacional (Aveiro, Portugal), 2000, pp. 343–356.

[Lea98] R.A.C.P. Leal, *Desenvolvimento de elementos finitos para análise de placas laminadas*, Vol. 6 (Coimbra, Portugal), Escola de Elementos Finitos, Centro Internacional de Matemática, 1998.

[LN94] M. Leung e G.E. Nevill, *Genetic algorithms for preliminary 2-d structural design*, AIAA 94 (1994), 2287–2291.

[LU97] LTH Lund University, *Calfem, a finite element toolbox to matlab, version 3.2*, Sweden, 1997.

[MM02] H. Mühlenbein e T. Mahnig, *Theoretical analysis of evolutionary computation*, Evolutionary Methods for Design, Optimization and Control (Barcelona, Spain) (J. Périaux K.D. Papailiou-T. Fogarty K.C. Giannakoglou, D.T. Tsahalis, ed.), CIMNE, 2002, pp. 27–34.

[NJG+96] S. Nagendra, D. Jestin, Z. Gürdal, R.T. Haftka, e L.T. Watson, *Improved genetic algorithms for the design of stiffened composite panels*, Computers & Structures (1996), no. 58, 543–555.

[NS96] S.G. Nash e A. Sofer, *Linear and nonlinear programming*, McGraw-Hill International Editions, New York, 1996.

[NW99] J. Nocedal e S.J. Wright, *Numerical optimization*, Springer-Verlag, New York, 1999.

[Pol97] C. Poloni, *Genetic algorithms in engineering and computer science*, ch. Hybrid GA for multiobjective aerodynamic shape optimization, p. 397, 1997.

[Rec64] I. Rechenberg, *Cybernetic solution path of an experimental problem*, Royal Aircraft Establishment (Farnborough, England), Library Translation No. 1122, 1964.

[Rec73] ———, *Evolutionsstrategie - optimierung technischer systeme nach prinzipien der biologischen evolution*, Frommann-Holzboog, Stuttgart, 1973.

[Rec94] ———, *Evolutionsstrategie '94*, Frommann-Holzboog, Stuttgart, 1994.

[RRR83] G.V. Reklaitis, A. Ravindran, e K.M. Ragsdell, *Engineering optimization*, ch. Methods and Applications, p. 277, 1983.

[RS95] H.S. Ryoo e B.P. Sahinidis, *Global optimization of nonconvex nlps and minlps with applications in process design*, Comp. chem. Engng. **19** (1995), no. 5, 551.

[Sal92] R.L. Salcedo, *Solving nonconvex nonlinear programming and mixed-integer nonlinear programming problems with adaptive random search*, Ind. Eng. Chem. Res. (1992), no. 31, 262.

[Sch81] H.-P. Schwefel, *Numerical optimization of computer models*, Wiley & Sons, Chichester, 1981.

[Sch85] J.D. Schaffer, *Multiple objective optimization with vector evaluated genetic algorithms*, Proceedings of the First International Conference on Genetic Algorithms (J.J. Grefensttete, ed.), Ed. Hillsdale, 1985, pp. 93–100.

[Sch95] H.-P. Schwefel, *Evolution and optimum seeking*, Wiley, New York, 1995.

[SD94] N. Srinivas e K. Deb, *Multi-objective function optimization using non-dominated sorting genetic algorithms*, Evolutionary Computation **2** (1994), no. 3, 221–248.

[SDJ91] W.M. Spears e K.A. De Jong, *On the virtues of parameterised uniform crossover*, Proceedings of the Fourth International Conference on Genetic Algorithms (San Mateo, California), Morgam Kaufmann Publishers, 1991, pp. 230–236.

[SNT85] Y. Sawaragi, H. Nakayama, e T. Tanino, *Theory of multiobjective optimization*, Academic Press, 1985.

[Sys89] G. Sysverda, *Uniform crossover in genetic algorithms*, Proceedings of the Third International Conference on Genetic Algorithms (San Mateo, California) (G.J.E. Rawlings, ed.), Morgam Kaufmann Publishers, 1989, pp. 2–9.

[Vos99] M.D. Vose, *The simple genetic algorithm: foundations and theory*, MIT Press, 1999.

[VVL00] D.A. Van Veldhuizen e G.B. Lamont, *Multiobjective evolutionary algorithms: Analysing the state-of-the-art*, Evolutionary Computation **8** (2000), no. 2, 125–147.

[Whi89] D. Whitley, *The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best*, Proceedings of the Third International Conference on Genetic Algorithms (San Mateo, California) (G. J. E. Rawlings, ed.), Morgam Kaufmann Publishers, 1989, pp. 116–121.

[Won90] J. Wong, *Computational experience with a general nonlinear programming algorithm*, COED J. **10** (1990), no. 19.

[Wri91] A.H. Wright, *Genetic algorithms for real parameter optimization*, Foundations of Genetic Algorithms (San Mateo, CA) (G.J.E. Rawlings, ed.), Morgam Kaufmann Publishers, 1991, pp. 205–218.

[YCL95] K.P. Yoon e Hwang C.-L., *Multiple attribute decision making: An introduction*, 104, vol. 07, Sage University Paper series on Quantitative Applications in the Social Sciences, Thousand Oak, CA: Sage, 1995.

[YZPD89] X. Yuan, S. Zhang, L. Pibouleau, e S. Domenech, *Une methode d'optimization nonlineaire en variables mixtes pour la conception de procedes*, RAIRO-Oper. Res. (1989).

[ZDT00] E. Zitzler, K. Deb, e L. Thiele, *Comparison of multiobjective evolutionary algorithms: Empirical results*, Evolutionary Computation **8** (2000), 173–195.

[Zit99] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: Methods and applications*, Ph.D. thesis, Swiss Federal Institute of Technology Zurich, 1999.

[ZT99] E. Zitzler e L. Thiele, *Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach*, IEEE Transactions on Evolutionary Computation **3** (1999), no. 4, 257–271.